Practical "Signatures with Efficient Protocols" from Simple Assumptions

Benoît Libert, Fabrice Mouhartem, Thomas Peters, Moti Yung

Practical "Signatures with Efficient Protocols" from Simple Assumptions

Benoît Libert
École Normale Supérieure de Lyon, France

Fabrice Mouhartem
École Normale Supérieure de Lyon, France

Thomas Peters*
Université Catholique de Louvain, Belgium

Moti Yung†
Snapchat & Columbia University, USA

ABSTRACT

Digital signatures are perhaps the most important base for authentication and trust relationships in large scale systems. More specifically, various applications of signatures provide privacy and anonymity preserving mechanisms and protocols, and these, in turn, are becoming critical (due to the recently recognized need to protect individuals according to national rules and regulations). A specific type of signatures called "signatures with efficient protocols", as introduced by Camenisch and Lysyanskaya (CL), efficiently accommodates various basic protocols and extensions like zero-knowledge proofs, signing committed messages, or re-randomizability. These are, in fact, typical operations associated with signatures used in typical anonymity and privacy-preserving scenarios. To date there are no "signatures with efficient protocols" which are based on simple assumptions and truly practical. These two properties assure us a robust primitive: First, simple assumptions are needed for ensuring that this basic primitive is mathematically robust and does not require special *ad hoc* assumptions that are more risky, imply less efficiency, are more tuned to the protocol itself, and are perhaps less trusted. In the other dimension, efficiency is a must given the anonymity applications of the protocol, since without proper level of efficiency the future adoption of the primitives is always questionable (in spite of their need). In this work, we present a new CL-type signature scheme that is re-randomizable under a simple, well-studied, and by now standard, assumption (SXDH). The signature is efficient (built on the recent QA-NIZK constructions), and is, by design, suitable to work in extended contexts that typify privacy settings (like anonymous credentials, group signature, and offline e-cash). We demonstrate its power by presenting practical protocols based on it.

*Work done while at ENS, Paris.
†Work done while in Google

Keywords: Signature; Signature with Efficient Protocols; Privacy; Anonymous Credentials; Group Signatures; Efficient Privacy-Enhancing Protocols; Simple Cryptographic Assumptions.

1. INTRODUCTION

Digital signatures are one of the fundamental cryptographic building blocks used in countless applications. The design of privacy-preserving protocols often requires signature schemes that are compatible with other primitives, primarily zero-knowledge proofs. Namely, it should be possible to sign messages without affecting their algebraic structure (in particular, without hashing them), in a way that maintains a signature holder's capability of efficiently proving statements about hidden message-signature pairs. Primitives like "structure-preserving signatures" [1, 39] or "signatures with efficient protocols" [18] were designed exactly in this spirit. While the former are motivated by security proofs in the standard model via the Groth-Sahai proof systems [32], the latter aim at enabling truly practical schemes.

Signature schemes with efficient protocols by Camenisch and Lysyanskaya [18] typically extend the functionalities of ordinary digital signatures in two ways: (i) They provide a protocol whereby a signer can obliviously sign a committed message known only to the user; (ii) Users should be able to efficiently prove possession of a hidden message-signature pair in a zero-knowledge manner. The latter property has proved extremely useful in the design of many efficient anonymity-related protocols such as anonymous credentials [21, 17], compact e-cash [16], revocable group signatures [43], oblivious transfer with access control [15] or certified private set intersection protocols [20].

The quality of a signature scheme is measured in two dimensions (and interactions thereof): first, the simplicity of the cryptographic assumption on which it is based, and secondly, its computational efficiency. So far, regarding cryptographic assumptions, most signature schemes with efficient protocols either require groups of hidden order [18] – where elements need a longer representation to keep the group order hidden – or they rely on non-standard hardness assumptions [19, 4, 45] in groups with a bilinear maps (or both since the Strong RSA assumption [5], which [18] relies on, is usually not recognized as standard). Camenisch and Lysyanskaya (CL) [19] showed how to adapt their Strong-RSA-based scheme in pairing-friendly groups. Their scheme, however, relies

on the interactive LRSW assumption [42]. Moreover, as pointed out in [46], it requires $O(n)$ group elements to sign messages made of ℓ blocks. Pointcheval and Sanders [46] recently modified CL signatures to sign ℓ-block messages using $O(1)$ group elements, but their scheme is only proven secure in the generic group model. While the first CL signature [18] has a natural counterpart [4, 45] based on a non-interactive assumption, it still requires a non-standard "q-type" assumption [13] where the number of input elements depends on the number of adversarial queries. We note that here we call all the above assumptions "non standard" (whether they are employed in the regular or random oracle models). For the time being, we are only aware of two schemes based on fixed-size assumptions. The first one is a variant, due to Gerbush *et al.* [33], of Camenisch-Lysyanskaya signatures [19] in composite order groups. Due to its much larger group order, it is inherently much less efficient than solutions in prime-order groups: for equivalent security levels, Freeman estimates [29] that computing a pairing over groups of order $N = pq$ is at least 50 times slower than the same pairing in the prime order setting. The second construction is a scheme, proposed by Yuen *et al.* [49] under the Decision Linear assumption [14]. Unfortunately, unlike LRSW-based Camenisch-Lysyanskaya signatures [19], it is deficient as it does not provide "randomizable signatures," an important property which – in the context of group signatures, for instance – enables re-randomization of credentials across distinct privacy-preserving authentications, and allows for a better efficiency.

Our Contribution. In this paper, we propose a new signature scheme with efficient protocols and re-randomizable signatures under simple, well-studied assumptions. The security of our scheme is proved in the standard model under the Symmetric eXternal Diffie-Hellman (SXDH) assumption, which is a well-established, constant-size assumption (i.e., described using a constant number of elements, regardless of the number of adversarial queries) in groups with a bilinear map. Remarkably, we can sign ℓ-block messages using only 4 group elements under the SXDH assumption.

Our signature length is enabled by the use of efficient Quasi-Adaptive Non-Interactive Zero-Knowledge (QA-NIZK) arguments for linear subspaces. As introduced by Jutla and Roy [34], QA-NIZK arguments are computationally sound proofs where the common reference string (CRS) may depend on the language of which membership must be proved. It was shown [40, 35, 38] that, for the task of arguing that a vector of group elements belongs to some linear subspace, the size of arguments may be independent of the dimensions of the considered subspace. Our signature scheme crucially exploits this observation as ℓ-block messages are signed by generating a QA-NIZK argument for a subspace of dimension $O(\ell)$.

Our signature natively supports efficient privacy-enhancing protocols. We describe a two-party protocol allowing a user to obtain a signature on a committed multi-block message as well as a honest-verifier zero-knowledge protocol for efficiently demonstrating knowledge of a signature on a committed message revealing neither the message nor the signature. Hence, our scheme readily enables the design of an efficient anonymous credentials system based on the sole SXDH assumption.

As another application of our signature scheme, we describe a truly practical group signature (for dynamic groups) based on simple assumptions in the random oracle model. Our scheme is competitive with the best solutions [14, 27] based on non-interactive assumptions (which are those relying on the Strong Diffie-Hellman assumption [13]) in terms of computational cost and signature length. Concretely, at the 128-bit security level, each signature fits within 320 bytes while providing anonymity in the strongest sense (i.e., against adversaries equipped with a signature opening oracle). To the best of our knowledge, the new scheme thus features the shortest group signatures based on standard assumptions.

It seems that our signature scheme has many other potential applications. For example, combining it with the ideas of [16] and a pseudo-random function based on standard assumptions (e.g., [44]) readily gives a compact e-cash system based on simple hardness assumptions.

Related Work. Anonymous credentials were introduced by Chaum [21] and efficiently designed by Camenisch and Lysyanskaya [17, 18]. They involve credential issuers and users who have a long-term secret key and pseudonyms which can be seen as commitments to their secret key. Users can obtain credentials from an issuer which only knows their pseudonym and obliviously certifies their secret key along with (optionally) a set of associated attributes. Users can subsequently interact with service providers – who know them under a different pseudonym – and demonstrate possession of the issuer's signature on their secret key without leaking anything else. Anonymous credentials involve a protocol allowing the user to obtain the issuer's signature on a committed message, a protocol for proving that two commitments open to the same message and a protocol for proving possession of a signature on a committed message.

Camenisch and Lysyanskaya gave the first efficient solutions based the Strong RSA assumption [17, 18]. Variants based on bilinear maps were considered in, e.g., [19, 2]. In the non-interactive setting (i.e., without interactive conversations between provers and verifiers) solutions in the standard model were given in [7, 6]. As a matter of fact, all truly practical solutions [18, 19, 2] require non-standard *ad hoc* assumptions.

Group signatures are a central privacy primitive, coined by Chaum and van Heyst [22], where members of a group managed by some authority can sign messages messages in the name of the group. Group member's accountability is enforced by means of an opening authority that can identify misbehaving signers. Ateniese, Camenisch, Joye and Tsudik [3] provided the first viable solution meeting the natural security requirements of the primitive, although rigorous security definitions were not available yet. These appeared later on in the work of Bellare, Micciancio and Warinschi [9], which [36, 11] subsequently extended to the dynamic setting. In these models, efficient schemes have been put forth in the random oracle model [36, 27] and the standard model [31]. As of now, however, a truly practical solution based on constant-size assumptions in the random oracle model remains lacking.

2. BACKGROUND

Notations. We let $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T$ denote groups of prime order $p \geq 2^\lambda$ where $\lambda \in \mathbb{N}$ is the security parameter. Bold capital letters will denote matrices, like $\mathbf{M}$, and

bold lowercase letters stand for vectors, like $\boldsymbol{v}$. Finally PPT stands for *probabilistic polynomial time*.

2.1 Hardness Assumptions

We use bilinear maps $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ over groups of prime order p where $e(g, \hat{h}) \neq 1_{\mathbb{G}_T}$ iff $g \neq 1_{\mathbb{G}}$ and $\hat{h} \neq 1_{\hat{\mathbb{G}}}$. We rely on hardness assumptions that are non-interactive and described using a constant number of elements.

Definition 1. The **Decision Diffie-Hellman** (DDH) problem in $\mathbb{G}$, is to distinguish the distributions (g^a, g^b, g^{ab}) and (g^a, g^b, g^c), with $a, b, c \xleftarrow{R} \mathbb{Z}_p$. The DDH assumption is the intractability of the problem for any PPT distinguisher.

The SXDH assumption posits the hardness of DDH in $\mathbb{G}$ and $\hat{\mathbb{G}}$, when $\mathbb{G} \neq \hat{\mathbb{G}}$.
We also rely on the following problem, which generalizes the Discrete Logarithm problem to asymmetric pairings.

Definition 2. In bilinear groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$ of prime order p, the **Symmetric Discrete Logarithm** (SDL) problem consists in, given $(g, \hat{g}, g^a, \hat{g}^a) \in \mathbb{G} \times \hat{\mathbb{G}}$ where $a \xleftarrow{R} \mathbb{Z}_p$, computing $a \in \mathbb{Z}_p$.

2.2 Quasi-Adaptive NIZK Arguments for Linear Subspaces

Quasi-Adaptive NIZK (QA-NIZK) proofs [34] are NIZK proofs where the common reference string (CRS) may depend on the language for which proofs have to be generated. Formal definitions are given in [34, 40, 38].
This section recalls the QA-NIZK argument of [38] for proving membership in the row space of a matrix. In the description below, we assume that all algorithms take as input the description of common public parameters cp consisting of asymmetric bilinear groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, p)$ of prime order $p > 2^\lambda$, where λ is the security parameter. In this setting the problem is to convince that $\boldsymbol{v}$ is a linear combination of the rows of a given $\mathbf{M} \in \mathbb{G}^{t \times n}$.
Kiltz and Wee [38] suggested the following construction which simplifies [40] and remains secure under SXDH. We stress that cp is independent of $\mathbf{M} = (\vec{M}_1 \cdots \vec{M}_t)^T$.

Keygen(cp, M): Given public parameters $\mathsf{cp} = (\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, p)$ and the matrix $M = (M_{i,j}) \in \mathbb{G}^{t \times n}$. Then, choose $\hat{g}_z \xleftarrow{R} \hat{\mathbb{G}}$. Pick $\mathsf{tk} = (\chi_1, \ldots, \chi_n) \xleftarrow{R} \mathbb{Z}_p^n$ and compute $\hat{g}_j = \hat{g}_z^{\chi_j}$, for all $j = 1$ to n. Then, for $i = 1$ to t, compute $z_i = \prod_{j=1}^n M_{i,j}^{-\chi_j}$ and output $\mathsf{crs} = \left(\{z_i\}_{i=1}^t,\ \hat{g}_z,\ \{\hat{g}_j\}_{j=1}^n\right) \in \mathbb{G}^t \times \hat{\mathbb{G}}^{n+1}$.

Prove$(\mathsf{crs}, \boldsymbol{v}, \{\boldsymbol{\omega_i}\}_{\boldsymbol{i=1}}^{\boldsymbol{t}})$: To prove that $\boldsymbol{v} = \vec{M}_1^{\omega_1} \cdots \vec{M}_t^{\omega_t}$, for some witness $\omega_1, \ldots, \omega_t \in \mathbb{Z}_p$, parse crs as above and return $\pi = \prod_{i=1}^t z_i^{\omega_i}$.

Sim$(\mathsf{tk}, \boldsymbol{v})$: In order to simulate a proof for a vector $\boldsymbol{v} \in \mathbb{G}^n$ using $\mathsf{tk} = \{\chi_i\}_{i=1}^n$, output $\pi = \prod_{j=1}^n v_j^{-\chi_j}$.

Verify$(\mathsf{crs}, \boldsymbol{v}, \boldsymbol{\pi})$: Given $\pi \in \mathbb{G}$ and $\boldsymbol{v} = (v_1, \ldots, v_n)$, return 1 if and only if $(v_1, \ldots, v_n) \neq (1_{\mathbb{G}}, \ldots, 1_{\mathbb{G}})$ and π satisfies $1_{\mathbb{G}_T} = e(\pi, \hat{g}_z) \cdot \prod_{j=1}^n e(v_j, \hat{g}_j)$.

The proof of the soundness of this QA-NIZK argument system requires the matrix $\mathbf{M}$ to be witness-samplable. This means that the reduction has to know the discrete logarithms of the group elements of $\mathbf{M}$. This requirement is compatible with our security proofs.

3. A RANDOMIZABLE SIGNATURE ON MULTI-BLOCK MESSAGES

In [41], Libert *et al.* described an F-unforgeable signature based on the SXDH assumption. We show that their scheme implies an efficient ordinary digital signature which makes it possible to efficiently sign multi-block messages in $\mathbb{Z}_p^\ell$ while keeping the scheme compatible with efficient protocols. In order to keep the signature length independent of the number of blocks, we exploit the property that the underlying QA-NIZK argument [38] has constant size, regardless of the dimensions of the considered linear subspace. Moreover, we show that their scheme remains unforgeable under the SXDH assumption.

Keygen(λ, ℓ) : Choose bilinear groups $\mathsf{cp} = (\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, p)$ of prime order $p > 2^\lambda$ with $g \xleftarrow{R} \mathbb{G}$, $\hat{g} \xleftarrow{R} \hat{\mathbb{G}}$.

1. Choose $\omega, a \xleftarrow{R} \mathbb{Z}_p$, and set $h = g^a$, $\Omega = h^\omega$.
2. Choose $\vec{v} = (v_1, \ldots, v_\ell, w) \xleftarrow{R} \mathbb{G}^{\ell+1}$.
3. Define a matrix $\mathbf{M} = (M_{j,i})_{j,i} \in \mathbb{G}^{(\ell+2) \times (2\ell+4)}$

$$\mathbf{M} = \left(\begin{array}{c|c|c|c} g & \mathbf{1}_{\ell+1} & \mathbf{1}_{\ell+1} & h \\ \hline \vec{v}^\top & g^{\mathbf{I}_{\ell+1}} & h^{\mathbf{I}_{\ell+1}} & \mathbf{1}_{\ell+1}^\top \end{array}\right), \qquad (1)$$

   where $\mathbf{1}_{\ell+1} = (1_{\mathbb{G}}, \ldots, 1_{\mathbb{G}}) \in \mathbb{G}^{\ell+1}$.
4. Run $\mathsf{Keygen}(\mathsf{cp}, M)$ of the QA-NIZK argument of Section 2.2 to get $\mathsf{crs} = (\{z_i\}_{i=1}^{\ell+2},\ \hat{g}_z,\ \{\hat{g}_j\}_{j=1}^{2\ell+4})$.

The private key is $\mathsf{sk} := \omega$ and the public key is

$$\mathsf{pk} = \Big(\mathsf{cp},\ g,\ h,\ \hat{g},\ \vec{v},\ \Omega = h^\omega,\ \mathsf{crs}\Big).$$

Sign$(\mathsf{sk}, \vec{m} = (m_1, \ldots, m_\ell))$: given the private key $\mathsf{sk} = \omega$ and a message $\vec{m} \in \mathbb{Z}_p^\ell$, choose $s \xleftarrow{R} \mathbb{Z}_p$ to compute

$$\sigma_1 = g^\omega \cdot (v_1^{m_1} \cdots v_\ell^{m_\ell} \cdot w)^s, \quad \sigma_2 = g^s, \quad \sigma_3 = h^s.$$

Then, run Prove of the QA-NIZK argument to prove that the following vector of $\mathbb{G}^{2\ell+4}$

$$(\sigma_1, \sigma_2^{m_1}, \ldots, \sigma_2^{m_\ell}, \sigma_2, \sigma_3^{m_1}, \ldots, \sigma_3^{m_\ell}, \sigma_3, \Omega) \qquad (2)$$

is in the row space of $\mathbf{M}$. This QA-NIZK proof $\pi \in \mathbb{G}$ consists of $\pi = z_1^\omega \cdot (z_2^{m_1} \cdots z_{\ell+1}^{m_\ell} \cdot z_{\ell+2})^s$.

Return the signature $\sigma = \big(\sigma_1, \sigma_2, \sigma_3, \pi\big) \in \mathbb{G}^4$.

Verify$(\mathsf{pk}, \sigma, \vec{m})$: parse σ as above and $\vec{m}$ as a tuple $(m_1, \ldots, m_\ell)$ in $\mathbb{Z}_p^\ell$ and return 1 if and only if

$$\begin{aligned} e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(\pi, \hat{g}_z) \cdot e(\sigma_1, \hat{g}_1) \\ \cdot e(\sigma_2, \hat{g}_2^{m_1} \cdots \hat{g}_{\ell+1}^{m_\ell} \cdot \hat{g}_{\ell+2}) \\ \cdot e(\sigma_3, \hat{g}_{\ell+3}^{m_1} \cdots \hat{g}_{2\ell+2}^{m_\ell} \cdot \hat{g}_{2\ell+3}). \end{aligned} \qquad (3)$$

The signature on ℓ scalars thus only consists of 4 elements in $\mathbb{G}$ while the verification equation only involves a computation of 5 pairings.

THEOREM 1. *The above signature scheme is existentially unforgeable under chosen-message attacks (eu-cma) if the SXDH assumption holds in* $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$. (The proof is available in Appendix A.)

4. COMPANION PROTOCOLS

In this section, we give Σ protocols for issuing a signature on a committed multi-block message and for proving knowledge of a valid message-signature pair.

4.1 Σ-Protocols

As defined by Cramer [23], Σ protocols are three-move honest-verifier zero-knowledge protocols where the first and last messages are sent by the prover.

Definition 3. [23] A Σ protocol for an NP language $\mathcal{L} : \{s \mid \exists w : L(s,w) = 1\}$ is a pair of interactive algorithms (P, V) that work as follows. On input (s, w) for P and s for V, the following interaction takes place:

1. P outputs a "commitment" com to the verifier.
2. V selects a "challenge" chall uniformly at random from a challenge space and sends it to the prover.
3. P sends a "response" resp and halts.

Eventually, V evaluates a predicate Verify on the statement s and the transcript $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$ and returns 0 or 1, then halts.

Beyond the completeness requirement (i.e., an honest run between $P(s,w)$ and $V(s)$ always accepts if $L(s,w) = 1$), the following security properties should be satisfied:

Special soundness A matching pair of transcripts w.r.t. a statement s is a pair $\mathsf{trans}_1 = (\mathsf{com}_1; \mathsf{chall}_1; \mathsf{resp}_1)$ and $\mathsf{trans}_2 = (\mathsf{com}_2; \mathsf{chall}_2; \mathsf{resp}_2)$ which are both accepting conversations, $\mathsf{com}_1 = \mathsf{com}_2$ but $\mathsf{chall}_1 \neq \mathsf{chall}_2$. A Σ protocol has special soundness if there is an extractor $\mathsf{Extract}$ that takes as input a statement s and a matching pair of transcripts $(\mathsf{trans}_1; \mathsf{trans}_2)$ and returns a witness w such that $L(s; w) = 1$.

Special honest verifier ZK A Σ protocol has special honest verifier zero knowledge (SHVZK) if there is a simulator Sim that takes as input a statement s (that may or may not be valid) and a challenge chall and outputs a transcript $(\mathsf{com}; \mathsf{chall}; \mathsf{resp})$ using the challenge provided such that $\mathsf{Verify}(s; (\mathsf{com}; \mathsf{chall}; \mathsf{resp})) = 1$. Furthermore, transcripts produced by the simulator for correct statements s are indistinguishable from transcripts produced by P and V on input s, where P additionally has any witness w for s as input.

Several techniques [26, 30] are known to transform Σ protocols (SHVZK) into interactive zero-knowledge proofs which remain secure against malicious verifiers.

4.2 Proof of Knowledge of a Signature on a Committed Message

We give Σ protocols for proving the knowledge of a signature-message pair $(\sigma, \vec{m})$ satisfying the verification equation of the scheme of Section 3

$$e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(\sigma_1, \hat{g}_1) \cdot e(\sigma_2, \hat{g}_2^{m_1} \cdots \hat{g}_{\ell+1}^{m_\ell} \cdot \hat{g}_{\ell+2}) \cdot e(\sigma_3, \hat{g}_{\ell+3}^{m_1} \cdots \hat{g}_{2\ell+2}^{m_\ell} \cdot \hat{g}_{2\ell+3}) \cdot e(\pi, \hat{g}_z), \quad (4)$$

where $\sigma = (\sigma_1, \sigma_2, \sigma_3, \pi)$ and $\vec{m} = (m_1, \ldots, m_\ell)$. We note that, as shown in the proof of Theorem 1, a candidate signature $(\sigma_1, \sigma_2, \sigma_3, \pi)$ may satisfy the verification equation although $\log_g(\sigma_2) \neq \log_h(\sigma_3)$. In applications to anonymous credentials, a malicious credential issuer could take advantage of this fact in attempts to break the anonymity of the scheme (e.g., by linking two authentications involving the same credential). For this reason, we consider a protocol for proving possession of a possibly maliciously generated signature.

We thus consider the case of arbitrary valid signatures that may have been maliciously computed by a signer who, e.g., aims at tracing provers across different authentications. In this setting, we can still obtain a perfect SHVZK Σ protocol to hedge against such attacks.

A first attempt to efficiently build such a protocol is to "linearize" the verification equation (4) by making sure that two witnesses are never paired together. However, we will still have to deal with (parallelizable) intermediate Σ protocols for quadratic scalar relations. Even though a quadratic pairing-product equation $e(x_1, \hat{a}) \cdot e(x_2, \hat{y})$ – for variables $x_1, x_2, \hat{y}$ and constant $\hat{a}$ – can be linearized by partially randomizing the variables so as to get the equation $e(x_1 \cdot x_2^r, \hat{a}) \cdot e(x_2, \hat{y} \cdot \hat{a}^{-r})$ (which allows $\hat{y}' = \hat{y} \cdot \hat{a}^{-r}$ to appear in the clear), proving knowledge of a valid signature still requires proving a statement about some representation of $\hat{y}$ which now appears in committed form. Somehow, going through the randomizing factor $\hat{a}^{-r}$ involves a quadratic relation between some known exponents to get special-soundness. To ease the entire proof we rather directly commit to the variables in $\mathbb{G}$ and $\hat{\mathbb{G}}$ using their available generator g and $\hat{g}$ which are not among the constants of the verification equation of the signature. We additionally need an extra generator f of $\mathbb{G}$ whose discrete logarithm is unknown.

Commit Given $(\sigma, \vec{m})$, conduct the following steps.

1. Commit to $d_1 := \hat{g}_2^{m_1} \cdots \hat{g}_{\ell+1}^{m_\ell} \cdot \hat{g}_{\ell+2} \in \hat{\mathbb{G}}$ and $d_2 := \hat{g}_{\ell+3}^{m_1} \cdots \hat{g}_{2\ell+2}^{m_\ell} \cdot \hat{g}_{2\ell+3} \in \hat{\mathbb{G}}$. To this end, choose $r_1, r_2 \xleftarrow{R} \mathbb{Z}_p$ and compute $\hat{D}_1 = d_1 \cdot \hat{g}^{r_1}$ and $\hat{D}_2 = d_2 \cdot \hat{g}^{r_2}$.
2. In order to prove knowledge of an opening of commitments $\hat{D}_1, \hat{D}_2 \in \hat{\mathbb{G}}$ to the same message $\vec{m} = (m_1, \ldots, m_\ell) \in \mathbb{Z}_p^\ell$, choose $s_1, s_2, u_1, \ldots, u_\ell \xleftarrow{R} \mathbb{Z}_p$ and compute $\hat{E}_1 = \hat{g}_2^{u_1} \cdots \hat{g}_{\ell+1}^{u_\ell} \cdot \hat{g}^{s_1}$ and $\hat{E}_2 = \hat{g}_{\ell+3}^{u_1} \cdots \hat{g}_{2\ell+2}^{u_\ell} \cdot \hat{g}^{s_2}$.
3. Using $r_1, r_2 \in \mathbb{Z}_p$ from step 1, define $\sigma_0 = \sigma_2^{r_1} \cdot \sigma_3^{r_2}$ and commit to $(\pi, \sigma_0, \sigma_1, \sigma_2, \sigma_3) \in \mathbb{G}^5$. For this purpose, choose $t_z, t_0, t_1, t_2, t_3 \xleftarrow{R} \mathbb{Z}_p$ at random and set $C_z = \pi \cdot g^{t_z}$, $C_i = \sigma_i \cdot g^{t_i}$, for $i \in \{0, \ldots, 3\}$, and $\hat{D}_0 = \hat{g}_z^{t_z} \cdot \hat{g}_1^{t_1} \cdot \hat{D}_1^{t_2} \cdot \hat{D}_2^{t_3} \cdot \hat{g}^{-t_0}$.
4. In order to prove (partial) knowledge of an opening to $(C_z, C_0, C_1, C_2, C_3, \hat{D}_0)$, compute $\hat{E}_0 = \hat{g}_z^{v_z} \cdot \hat{g}_1^{v_1} \cdot \hat{D}_1^{v_2} \cdot \hat{D}_2^{v_3} \cdot \hat{g}^{-v_0}$ for random $v_z, v_0, v_1, v_2, v_3 \xleftarrow{R} \mathbb{Z}_p$.
5. Prove that C_0 is well-formed relatively to the committed values in C_1, C_2 and the coins $r_1, r_2 \in \mathbb{Z}_p$ used in $\hat{D}_1, \hat{D}_2$. To this end, prove knowledge of the representation $C_0 = C_2^{r_1} \cdot C_3^{r_2} \cdot g^{t_4}$, where $t_4 = t_0 - r_1 \cdot t_2 - r_2 \cdot t_3$. To do this, compute $F_0 = C_2^{s_1} \cdot C_3^{s_2} \cdot g^{v_4}$, for $v_4 \xleftarrow{R} \mathbb{Z}_p$ and where $s_1, s_2 \in \mathbb{Z}_p$ are the random coins used in $\hat{E}_1, \hat{E}_2$.
6. To prove that $t_4 = t_0 - r_1 \cdot t_2 - r_2 \cdot t_3$, (re-)commit to $t_0, t_2, t_3, t_4 \in \mathbb{Z}_p$ by picking $x_2, x_3, x_4 \xleftarrow{R} \mathbb{Z}_p$ and computing

$$T_i = g^{t_i} \cdot f^{x_i} \qquad \forall i \in \{0, 2, 3, 4\},$$

where $x_0 = x_2 \cdot r_1 + x_3 \cdot r_2 + x_4$. Ensure that committed variables coincide with those of previous steps by computing

$$\{V_i = g^{v_i} \cdot f^{y_i}\}_{i\in\{0,2,3,4\}},$$

where $y_0, y_2, y_3, y_4 \xleftarrow{R} \mathbb{Z}_p$. To prove the equality $T_0 = T_2^{r_1} \cdot T_3^{r_2} \cdot T_4$, re-use $s_1, s_2 \in \mathbb{Z}_p$ from steps 2 and 5 to compute $S_0 = T_2^{s_1} \cdot T_3^{s_2}$.

Finally, keep $C_z \in \mathbb{G}$ and all the random coins in aux, and output

$$\mathsf{com} = \Big(\{C_i\}_{i=0}^{3}, F_0, \{(T_i, V_i)\}_{i=0,2,3,4}, S_0, \{(\hat{D}_i, \hat{E}_i)\}_{i=0}^{2}\Big) \in \mathbb{G}^{14} \times \hat{\mathbb{G}}^{6} \quad (5)$$

Challenge Given com as per (5), pick $\rho \xleftarrow{R} \mathbb{Z}_p$ uniformly at random and return $\mathsf{chall} = \rho$.

Response On inputs com, aux and $\mathsf{chall} = \rho$, compute:

1. $\bar{m}_i = \rho \cdot m_i + u_i$, for $i = 1$ to ℓ, $\bar{r}_1 = \rho \cdot r_1 + s_1$, and $\bar{r}_2 = \rho \cdot r_2 + s_2$;
2. $w_z = \rho \cdot t_z + v_z$ and $w_i = \rho \cdot t_i + v_i$, for $i = 0$ to 3;
3. $w_4 = \rho \cdot t_4 + v_4$, where $t_4 := t_0 - t_1 \cdot r_1 - t_2 \cdot r_2$;
4. $z_i = \rho \cdot x_i + y_i$ for each $i \in \{0, 2, 3, 4\}$.

Output $\mathsf{resp} \in \mathbb{G} \times \mathbb{Z}_p^{\ell+12}$ as

$$\big(C_z, \{\bar{m}_i\}_{i=1}^{\ell}, \bar{r}_1, \bar{r}_2, w_z, \{w_i\}_{i=0}^{4}, \{z_i\}_{i=0,2,3,4}\big).$$

Verify Given $(\mathsf{com}; \mathsf{chall}; \mathsf{resp})$ return 0 if it does not parse correctly or if the following relations do not hold:

1. $(\hat{D}_1/\hat{g}_{\ell+2})^{\rho} \cdot \hat{E}_1 = \hat{g}_2^{\bar{m}_1} \cdots \hat{g}_{\ell+1}^{\bar{m}_\ell} \cdot g^{\bar{r}_1}$ and $(\hat{D}_2/\hat{g}_{2\ell+3})^{\rho} \cdot \hat{E}_2 = \hat{g}_{\ell+3}^{\bar{m}_1} \cdots \hat{g}_{2\ell+2}^{\bar{m}_\ell} \cdot g^{\bar{r}_2}$;
2. $\hat{D}_0^{\rho} \cdot \hat{E}_0 = \hat{g}_z^{w_z} \cdot \hat{g}_1^{w_1} \cdot \hat{D}_1^{w_2} \cdot \hat{D}_2^{w_3} \cdot \hat{g}^{-w_0}$ and $C_0^{\rho} \cdot F_0 = C_2^{\bar{r}_1} \cdot C_3^{\bar{r}_2} \cdot g^{w_4}$.
3. $T_i^{\rho} \cdot V_i = g^{w_i} f^{z_i}$ for each $i \in \{0, 2, 3, 4\}$ and
$$(T_0/T_4)^{\rho} \cdot S_0 = T_2^{\bar{r}_1} \cdot T_3^{\bar{r}_2}. \quad (6)$$

Then, return 1 if and only if

$$e(C_0, \hat{g}) \cdot e(g, \hat{D}_0) \cdot e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(C_1, \hat{g}_1) \cdot e(C_2, \hat{D}_1) \cdot e(C_3, \hat{D}_2) \cdot e(C_z, \hat{g}_z). \quad (7)$$

It is worth noticing that no pairing evaluation is required until the final step of Verify, which is almost as efficient as the verification of underlying signatures. Moreover, the prover's first message com is of constant-size and the communication complexity of the protocol exceeds the length of the witness by a constant additive overhead.

THEOREM 2. *The above interactive scheme is a secure Σ protocol for the language L_{sig} induced by the relation $R_{sig}(\mathsf{pk}, (\vec{\sigma}, \vec{m})) = 1$ if and only if $\mathsf{Verify}'(\mathsf{pk}, \vec{\sigma}, \vec{m}) = 1$, where $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}')$ is the signature of Section 3.*

PROOF. *Correctness.* Expanding an honestly generated $\hat{D}_0 = \hat{g}_z^{t_z} \cdot \hat{g}_1^{t_1} \cdot \hat{D}_1^{t_2} \cdot \hat{D}_2^{t_3} \cdot \hat{g}^{-t_0}$ in equation (7) and regrouping the pairing factors gives

$$e(C_0 \cdot g^{-t_0}, \hat{g}) \cdot e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(C_1 \cdot g^{-t_1}, \hat{g}_1) \cdot e(C_2 \cdot g^{-t_2}, \hat{D}_1) \cdot e(C_3 \cdot g^{-t_3}, \hat{D}_2) \cdot e(C_z \cdot g^{-t_z}, \hat{g}_z).$$

Now, expanding the commitments to group elements in $\mathbb{G}$ reduces this equation to

$$e(\sigma_2^{r_1} \cdot \sigma_3^{r_2}, \hat{g}) \cdot e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(\sigma_1, \hat{g}_1) \cdot e(\sigma_2, \hat{D}_1) \cdot e(\sigma_3, \hat{D}_2) \cdot e(\pi, \hat{g}_z)$$

which holds true for valid witnesses when $\hat{D}_1 = d_1 \cdot \hat{g}^{r_1}$ and $\hat{D}_2 = d_2 \cdot \hat{g}^{r_2}$. Remaining verifications of items 1,2,3 follow from the correctness of the built-in Σ protocols.

Special-Soundness. Let us assume two accepting transcripts $(\mathsf{com}, \rho, \mathsf{resp})$, $(\mathsf{com}, \rho', \mathsf{resp}')$ with $\rho \neq \rho'$. The special soundness of the sub-protocols involving $\hat{D}_1, \hat{D}_2$ (with $\hat{E}_1, \hat{E}_2$) – consisting of steps 1 and 2 of Commit and step 1 of Verify – ensures the extraction of $m_1, \ldots, m_\ell, r_1, r_2$ satisfying $\hat{D}_1 = d_1 \cdot \hat{g}^{r_1}$, where $d_1 = \hat{g}_2^{m_1} \cdots \hat{g}_{\ell+1}^{m_\ell} \cdot \hat{g}_{\ell+2}$, and $\hat{D}_2 = d_2 \cdot \hat{g}^{r_2}$, where $d_2 = \hat{g}_{\ell+3}^{m_1} \cdots \hat{g}_{2\ell+2}^{m_\ell} \cdot \hat{g}_{2\ell+3}$. From step 2 of Verify, a similar argument on $\hat{D}_0$ (with $\hat{E}_0$) implies the extractability of $(t_z, t_0, t_1, t_2, t_3, t_4)$ such that $\hat{D}_0 = \hat{g}_z^{t_z} \cdot \hat{g}_1^{t_1} \cdot \hat{D}_1^{t_2} \cdot \hat{D}_2^{t_3} \cdot \hat{g}^{-t_0}$. Moreover, together with previously extracted (r_1, r_2), step 2 of Verify also guarantees that t_4 satisfies $C_0 = C_2^{r_1} \cdot C_3^{r_2} \cdot g^{t_4}$.

We now state that quantities $\{\sigma_i = C_i \cdot g^{-t_i}\}_{i\in\{1,2,3\}}$ and $\pi = C_z \cdot g^{-t_z}$ satisfy (3), so that, together with $\vec{m} = (m_1, \ldots, m_\ell)$, they form a valid witness for R_{sig}. Namely, $(\sigma, \vec{m}) = ((\sigma_1, \sigma_2, \sigma_3, \pi), (m_1, \ldots, m_\ell))$ is a valid message-signature pair.

To see this, define $\sigma_0 = C_0 \cdot g^{-t_0}$. Since equation (7) holds by hypothesis, if we expand all commitments using extracted values, we find

$$e(\sigma_0, \hat{g}) \cdot e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(\sigma_1, \hat{g}_1) \cdot e(\sigma_2, d_1 \cdot \hat{g}^{r_1}) \cdot e(\sigma_3, d_2 \cdot \hat{g}^{r_2}) \cdot e(\pi, \hat{g}_z).$$

We are thus left with showing that $\sigma_0 = \sigma_2^{r_1} \cdot \sigma_3^{r_2}$ or, equivalently, $e(\sigma_0, \hat{g}) = e(\sigma_2, \hat{g}^{r_1}) \cdot e(\sigma_3, \hat{g}^{r_2})$. Remember that, from step 2 of Verify, we know that extracted $(r_1, r_2, t_4) \in \mathbb{Z}_p^3$ form a representation of C_0 w.r.t. the base (C_0, C_2, g): i.e., $C_0 = C_2^{r_1} \cdot C_3^{r_2} \cdot g^{t_4}$, which, from the definition of $\sigma_0, \sigma_2, \sigma_3$, yields $\sigma_0 \cdot g^{t_0} = \sigma_2^{r_1} \cdot \sigma_3^{r_2} \cdot g^{t_2 \cdot r_1 + t_3 \cdot r_2 + t_4}$. Hence, we are done if we can show that $t_0 = t_2 r_1 + t_3 r_2 + t_4$. But this exactly what step 3 of Verify and the special soundness of the sub-protocol involving (T_0, T_2, T_3, T_4) tells us. First, we have a representation of these T_i's w.r.t. the basis $(g, f) \in \mathbb{G}^2$ which guarantees that we are working on the already extracted (t_0, t_2, t_3, t_4) involved in the expressions of $\hat{D}_0$ and C_0. Second, the verification equation (6) ensures that $T_0 = T_2^{r_1} \cdot T_3^{r_2} \cdot T_4$ and the final result follows by replacing them by their representation.

Perfect SHVZK. To show this property we must build a simulator that, on input of a challenge $\mathsf{chall} = \rho \in_R \mathbb{Z}_p$, emulates a valid transcript without any witness. First, we need to compute a random tuple $C_z, \{C_i\}_{i=0}^{3}, \{\hat{D}\}_{i=0}^{2}$ constrained to satisfy the verification equation (7).

From the identity $e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(\Omega^{-1}, \hat{g}_{2\ell+4})$ we first pick $a_0, a_1, a_2, a_z \leftarrow \mathbb{Z}_p$, $\hat{D}_1 \leftarrow \hat{\mathbb{G}}$ and we have $e(\Omega, \hat{g}_{2\ell+4})^{-1} = e(\Omega^{-1}, \hat{g}_{2\ell+4} \cdot \hat{g}^{a_0} \hat{g}_1^{a_1} \hat{D}_1^{a_2} \hat{g}_z^{a_z}) \cdot e(\Omega^{a_0}, \hat{g}) \cdot e(\Omega^{a_1}, \hat{g}_1) \cdot e(\Omega^{a_2}, \hat{D}_1) \cdot e(\Omega^{a_z}, \hat{g}_z)$, so that we can set $C_0 = \Omega^{-a_0}$, $C_1 = \Omega^{a_1}$, $C_2 = \Omega^{a_2}$ and $C_z = \Omega^{a_z}$. Let $\hat{B} := \hat{g}_{2\ell+4} \cdot \hat{g}^{a_0} \hat{g}_1^{a_1} \hat{D}_1^{a_2} \hat{g}_z^{a_z}$. Now, we can introduce the constant $g \in \mathbb{G}$ in the equation by picking $a_g \leftarrow \mathbb{Z}_p$ since $e(\Omega^{-1}, \hat{B}) = e(\Omega^{-1} \cdot g^{a_g}, \hat{B}) \cdot e(g, \hat{B}^{-a_g})$. Then, we finally set $\hat{D}_0 = \hat{B}^{a_g}$, $\hat{D}_2 = \hat{B}^{a_3}$ and $C_3 = (\Omega^{-1} \cdot g^{a_g})^{1/a_3}$ for a random $a_3 \leftarrow \mathbb{Z}_p$.

To complete the simulated transcript, we run a parallel execution of the simulators of all Σ protocols used as subroutines. More explicitly, first pick $\rho \xleftarrow{R} \mathbb{Z}_p$ and $\bar{m}_1, \ldots, \bar{m}_\ell, \bar{r}_1, \bar{r}_2, w_z, w_0, \ldots, w_4, z_0, z_2, z_3, z_4 \xleftarrow{R} \mathbb{Z}_p$. Also, choose $T_0, T_2, T_3, T_4 \xleftarrow{R} \mathbb{G}$ and do the following:

1. Compute $\hat{E}_1 = (\hat{D}_1/\hat{g}_{\ell+2})^{-\rho} \cdot \hat{g}_2^{\bar{m}_1} \cdots \hat{g}_{\ell+1}^{\bar{m}_\ell} \cdot g^{\bar{r}_1}$ and, similarly, $\hat{E}_2 = (\hat{D}_2/\hat{g}_{2\ell+3})^{-\rho} \cdot \hat{g}_{\ell+3}^{\bar{m}_1} \cdots \hat{g}_{2\ell+2}^{\bar{m}_\ell} \cdot g^{\bar{r}_2}$;

2. Compute $F_0 = C_0^{-\rho} \cdot C_2^{\bar{r}_1} \cdot C_3^{\bar{r}_2} \cdot g^{w_4}$ as well as $\hat{E}_0 = \hat{D}_0^{\rho} \cdot \hat{g}_z^{w_z} \cdot \hat{g}_1^{w_1} \cdot \hat{D}_1^{w_2} \cdot \hat{D}_2^{w_3} \cdot \hat{g}^{-w_0}$;

3. Compute $V_i = T_i^{-\rho} \cdot g^{v_i} f^{z_i}$, for each $i \in \{0, 2, 3, 4\}$, and $S_0 = (T_0/T_4)^{-\rho} \cdot T_2^{\bar{r}_1} \cdot T_3^{\bar{r}_2}$.

This concludes the proof. □

4.3 Signing a Committed Message

At a high level, the protocol involves a committer who wants to get a signature on $\mathbf{m} = (m_1, \ldots, m_\ell)$ and first computes a commitment of the form $c_v = v_1^{m_1} \cdots v_\ell^{m_\ell} \cdot u^r$, where u is the extra public parameter (with unknown discrete log). The signer gives back elements of the form $\tau_1 = g^\omega c_v^s$, $\tau_2 = g^s$, $\tau_3 = h^s$ which is almost the desired signature. To get the component σ_1 of the right form relatively to τ_2, τ_3 the committer has to remove the factor u^{rs} from τ_1. Then, the signer also sends $\tau_0 = u^s$ to enable removing τ_0^r. In the protocol some randomizing steps are included as well as other additional components allowing the committer to extract π, the QA-NIZK part of the signature. In the security proof of the protocol we thus have to show that the additional value $\tau_0 = u^s$ does not affect the unforgeability of the signature.

The protocol. At the beginning of a new run of the protocol, the committer has a vector $\mathbf{m} = (m_1, \ldots, m_\ell)$, the public-key of the signature scheme and the extra generator $u \in \mathbb{G}$ (which can be a hashed point), the signer also has the secret key of the signature scheme but not $\mathbf{m}$. To get a signature on $\mathbf{m}$, the committer picks $r \xleftarrow{R} \mathbb{Z}_p$ and computes a perfectly hiding commitment $c_v = v_1^{m_1} \cdots v_\ell^{m_\ell} \cdot u^r \in \mathbb{G}$. Besides, it also computes the elements $c_z = z_2^{m_1} \cdots z_{\ell+1}^{m_\ell} \cdot u^{t_z}$. The signer receives these commitments and they both engage in an interactive proof of knowledge of an equal representation of c_v relatively to the basis $(v_1, \ldots, v_\ell; u)$ and c_z relatively to the basis $(z_2, \ldots, z_{\ell+1}; u)$, where the signer plays the role of the verifier. Depending on the success of the proof the signer computes what we can call a "pre-signature" consisting of the following group elements

$$\tau_1 = g^\omega \cdot (c_v \cdot w)^s, \quad \tau_3 = h^s, \quad \pi_0 = z_1^\omega \cdot c_z^s \cdot z_{\ell+2}^s,$$
$$\tau_2 = g^s, \quad \tau_0 = u^s,$$

for a random $s \xleftarrow{R} \mathbb{Z}_p$. In the final step, the user received the pre-signature, then picks $s' \xleftarrow{R} \mathbb{Z}_p$ and computes $(\sigma_1, \sigma_2, \sigma_3, \pi) \in \mathbb{G}^4$ as follows

$$\sigma_1 = \tau_1 \cdot \tau_0^{-r} \cdot (v_1^{m_1} \cdots v_\ell^{m_\ell} \cdot w)^{s'}, \qquad \sigma_2 = \tau_2 \cdot g^{s'},$$
$$\pi = \pi_0 \cdot \tau_0^{-t_z} \cdot (z_2^{m_1} \cdots z_{\ell+1}^{m_\ell} \cdot z_{\ell+2})^{s'}, \quad \sigma_3 = \tau_3 \cdot h^{s'}.$$

Finally the user checks the validity of the signature. Depending on the validity, the user outputs the signature or a failure symbol $\perp$.

We notice that the number of transmitted group elements is constant and no pairing is needed before the signature verification phase. In comparison, the construction of [18] requires groups of larger hidden order and their protocol for signing committed message blocks requires a linear number of range proofs.

Security. We briefly sketch the proof of the above protocol in front of malicious entities since classical arguments can be applied. Assuming that the committer uses secure ZKPK and does not output $\perp$, a malicious signer which receives perfectly hiding commitments c_v, c_z cannot tell apart an honest proof from a simulated proof. Consequently the signer learns nothing from $\mathbf{m}$ during the execution of the protocol. In the other case, we have to show that a corrupted committer remains unable to produce valid signature on a new vector $\mathbf{m}^\star$. First, since the generation of u is not under the controlled of the committer but of the random oracle, u can be made independent of rest of pk. Then, we only need to show that the signature remains unforgeable when τ_0 is given in the signature. Since $\mathbf{m}$ and s can be extracted from the proof of knowledge the reduction can output a signature on $\mathbf{m}$. Moreover it is easy to see from the security proof (in Appendix A) of the signature how this additional element can be simulated. Actually the only place in the reduction where τ_0 could not be computed directly as u^s for a known s is when the challenger $\mathcal{B}$ has to embed an SXDH challenge in a simulated signature. Given (g, h, g^b, h^{b+c}), $\mathcal{B}$ can compute $u = g^{a_u} h^{b_u}$ from random $a_u, b_u \leftarrow \mathbb{Z}_p$ and program the random oracle to output this element u as the specification of the public-key would do. Then to simulate τ_0 $\mathcal{B}$ simply has to compute $\tau_0 = (g^b)^{a_u} (h^{b+c})^{b_v} = u^b h^{c \cdot b_v}$ which is u^b or random. The rest of the reduction remains unchanged since the value a_u, b_u are completely independent of those already described in the sketch of proof in Appendix A.

Remark. Since a malicious signer may know the simulation trapdoor $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$ of the underlying QA-NIZK argument, he could produce valid signature so that $\log_g \sigma_2 \neq \log_h \sigma_3$. Then, if the committer later needs to proof knowledge of the received signature it then has to use the sigma protocol of Section 3 where both σ_2 and σ_3 only appear in committed form.

5. DYNAMIC GROUP SIGNATURES

We adapt the protocol of section 3 to build a dynamic group signature [11, 36]. At a high level, each group member obtains a membership certificate consisting of a signature $(\sigma_1, \sigma_2, \sigma_3, \pi)$ on a message $\mathsf{ID} \in \mathbb{Z}_p$ which is only known to the group member. During the joining protocol, each group member thus obtains a signature on a committed message $\mathsf{ID} \in \mathbb{Z}_p$. Here, we use a deterministic commitment to ID, which suffices to ensure

security against framing attacks and allows for a better efficiency. When signing a message, each group member verifiably encrypts the components (σ_1, π) of his membership certificate that depend on ID (and not σ_2, σ_3 which can be assumed to be honestly computed here, unlike in the previous section). For the sake of efficiency, we use a randomness re-using [8] variant of the Cramer-Shoup encryption scheme [25] whereby σ_1 and π are both encrypted using the same encryption exponent $\theta \in \mathbb{Z}_p$. For public verifiability purposes, the validity of Cramer-Shoup ciphertexts is demonstrated using Σ protocols and the Fiat-Shamir heuristic [28] (somewhat in the fashion of [48]) rather than designated verifier NIZK proofs [25].

In the join protocol, the user proves knowledge of his membership secret $\mathsf{ID} \in \mathbb{Z}_p$ in a zero-knowledge manner, which restricts the group manager to sequentially interact with prospective users. However, this limitation can be removed using an extractable commitment as in [27].

Keygen(λ, N)**:** given $\lambda \in \mathbb{N}$, and the maximum number of users $N \in \mathsf{poly}(\lambda)$, choose asymmetric bilinear groups $\mathsf{cp} = (\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, p)$ of order $p > 2^\lambda$.

1. Generate a key pair $(\mathsf{pk}_s, \mathsf{sk}_s)$ for the scheme of section 3 for a one-block message (i.e., $\ell = 1$). The secret key is $\mathsf{sk}_s = \omega$, while the public key is
$$\mathsf{pk}_s = \big(\mathsf{cp},\ g,\ h,\ \hat{g},\ \vec{v} = (v, w),\ \Omega = h^\omega,\ \mathsf{crs}\big),$$
where $\mathsf{crs} = \big(\{z_j\}_{j=1}^3, \hat{g}_z, \{\hat{g}_i\}_{i=1}^6\big)$.
2. Pick $x_z, y_z, x_\sigma, y_\sigma, x_{\mathsf{ID}}, y_{\mathsf{ID}} \xleftarrow{R} \mathbb{Z}_p$ and set
$$X_z = g^{x_z} h^{y_z}, \quad X_\sigma = g^{x_\sigma} h^{y_\sigma}, \quad X_{\mathsf{ID}} = g^{x_{\mathsf{ID}}} h^{y_{\mathsf{ID}}}.$$
3. Choose a hash function $H : \{0,1\}^* \times \mathbb{G}^{10} \times \mathbb{G}_T \to \mathbb{Z}_p$ that will be modeled as a random oracle.
4. Define $\mathcal{Y} = \big\{\mathsf{pk}_s, X_z, X_\sigma, X_{\mathsf{ID}}\big\}$ to be the group public key. The group manager's private key is $\mathcal{S}_{\mathsf{GM}} = \omega = \mathsf{sk}_s$ whereas the opening authority's private key consists of $\mathcal{S}_{\mathsf{OA}} = \big(x_z, y_z, x_\sigma, y_\sigma, x_{\mathsf{ID}}, y_{\mathsf{ID}}\big)$.

Join$^{(\mathsf{GM},\mathcal{U}_i)}$**:** The group manager GM, and the prospective user $\mathcal{U}_i$ run the following interactive protocol:

1. $\mathcal{U}_i$ chooses $\mathsf{ID} \xleftarrow{R} \mathbb{Z}_p$ and sends the following to GM: $(V_{\mathsf{ID}}, Z_{\mathsf{ID}}, \hat{G}_{2,\mathsf{ID}}, \hat{G}_{4,\mathsf{ID}}) = (v^{\mathsf{ID}}, z_2^{\mathsf{ID}}, \hat{g}_2^{\mathsf{ID}}, \hat{g}_4^{\mathsf{ID}})$
2. GM checks that V_{ID} does not appear in any transcript of St and abort if it does. Otherwise (i.e., if V_{ID} is fresh), GM verifies that: for $k = 2, 4$,
$$e(V_{\mathsf{ID}}, \hat{g}) \stackrel{?}{=} e(g, \hat{G}_{k,\mathsf{ID}}), \quad e(Z_{\mathsf{ID}}, \hat{g}) \stackrel{?}{=} e(g, \hat{G}_{2,\mathsf{ID}}).$$
If all tests pass, samples a fresh index $i \in \mathbb{Z}_p$ and sends it to $\mathcal{U}_i$, otherwise abort.
3. $\mathcal{U}_i$ runs an interactive zero-knowledge proof of knowledge of $\mathsf{ID} = \log_v(V_{\mathsf{ID}})$ in interaction with GM. For instance, the 4-round protocol of Cramer *et al.* [24] can be used for this purpose. Let $\pi_K(\mathsf{ID})$ denote the interaction transcript.
4. GM uses $V_{\mathsf{ID}} = v^{\mathsf{ID}}$ to sign ID using the scheme of section 3: i.e., GM picks $s \xleftarrow{R} \mathbb{Z}_p$, and uses $\mathcal{S}_{\mathsf{GM}} = \omega$ to compute $\sigma_1 = g^\omega \cdot (V_{\mathsf{ID}} w)^s = g^\omega \cdot (v^{\mathsf{ID}} \cdot w)^s$ and
$$\sigma_2 = g^s, \qquad \sigma_3 = h^s.$$
Then GM uses Z_{ID} to generate the QA-NIZK proof $\pi \in \mathbb{G}$ as
$$\pi = z_1^\omega \cdot (Z_{\mathsf{ID}} \cdot z_3)^s = z_1^\omega \cdot (z_2^{\mathsf{ID}} \cdot z_3)^s$$
and finally sends $\mathsf{cert}_i = (i, V_{\mathsf{ID}}, \sigma_1, \sigma_2, \sigma_3, \pi)$
5. Finally GM and $\mathcal{U}_i$ respectively store
$$\mathsf{transcript}_i = \Big(\big(Z_{\mathsf{ID}}, \hat{G}_{2,\mathsf{ID}}, \hat{G}_{4,\mathsf{ID}}\big), \pi_K(\mathsf{ID}), \mathsf{cert}_i\Big) \quad (8)$$
and $(\mathsf{cert}_i, \mathsf{sec}_i) = \big((i, V_{\mathsf{ID}}, \sigma_1, \sigma_2, \sigma_3, \pi), \mathsf{ID}\big)$.

Sign$(\mathcal{Y}, \mathsf{sec}_i, \mathsf{cert}_i, M)$**:** Given a message $M \in \{0,1\}^*$ and a secret $\mathsf{sec}_i = \mathsf{ID}$, the user $\mathcal{U}_i$ does the following:

1. Re-randomize the certificate cert_i. Namely, choose $r \xleftarrow{R} \mathbb{Z}_p$ and compute $\tilde{\sigma}_2 = \sigma_2 \cdot g^r$, $\tilde{\sigma}_3 = \sigma_3 \cdot h^r$, $\tilde{\sigma}_1 = \sigma_1 \cdot (v^{\mathsf{ID}} \cdot w)^r$, $\tilde{\pi} = \pi \cdot (z_2^{\mathsf{ID}} \cdot z_3)^r$.
2. Encrypt elements $\tilde{\pi}$, $\tilde{\sigma}_1$ and v^{ID} from the membership certificate. Specifically, choose $\theta \xleftarrow{R} \mathbb{Z}_p$ and compute the Cramer-Shoup ciphertext $C_{\mathsf{CS}} = (C_1, C_2, C_z, C_\sigma, C_{\mathsf{ID}})$, where $C_1 = g^\theta$, $C_2 = h^\theta$,
$$C_z = \tilde{\pi} \cdot X_z^\theta, \quad C_\sigma = \tilde{\sigma}_1 \cdot X_\sigma^\theta, \quad C_{\mathsf{ID}} = v^{\mathsf{ID}} \cdot X_{\mathsf{ID}}^\theta.$$
3. Then, prove knowledge of $(\mathsf{ID}, \theta) \in \mathbb{Z}_p^2$ such that
$$C_1 = g^\theta, \qquad C_2 = h^\theta, \qquad C_{\mathsf{ID}} = v^{\mathsf{ID}} \cdot X_{\mathsf{ID}}^\theta,$$
$$\big(e(C_z, \hat{g}_z) \cdot e(C_\sigma, \hat{g}_1) \cdot e(\tilde{\sigma}_2, \hat{g}_3) \cdot e(\tilde{\sigma}_3, \hat{g}_5)\big)$$
$$= \big(e(X_z, \hat{g}_z) \cdot e(X_\sigma, \hat{g}_1)\big)^\theta \cdot \big(e(\tilde{\sigma}_2, \hat{g}_2) \cdot e(\tilde{\sigma}_3, \hat{g}_4)\big)^{-\mathsf{ID}}.$$
Namely, sample random $r_{\mathsf{ID}}, r_\theta \xleftarrow{R} \mathbb{Z}_p$, compute
$$R_1 = g^{r_\theta}, \quad R_2 = h^{r_\theta}, \quad R_3 = v^{r_{\mathsf{ID}}} \cdot X_{\mathsf{ID}}^{r_\theta},$$
$$R_4 = \big(e(X_z, \hat{g}_z) \cdot e(X_\sigma, \hat{g}_1)\big)^{r_\theta} \cdot \big(e(\tilde{\sigma}_2, \hat{g}_2) \cdot e(\tilde{\sigma}_3, \hat{g}_4)\big)^{-r_{\mathsf{ID}}}$$
and then $c = H(M, C_{\mathsf{CS}}, \tilde{\sigma}_2, \tilde{\sigma}_3, R_1, R_2, R_3, R_4)$. Finally compute $s_\theta = r_\theta + c \cdot \theta$, $s_{\mathsf{ID}} = r_{\mathsf{ID}} + c \cdot \mathsf{ID}$ in $\mathbb{Z}_p$.
4. Return the signature Σ which consists of
$$\Sigma = (C_{\mathsf{CS}}, \tilde{\sigma}_2, \tilde{\sigma}_3, c, s_{\mathsf{ID}}, s_\theta) \in \mathbb{G}^7 \times \mathbb{Z}_p^3 \quad (9)$$

Verify$(\mathcal{Y}, M, \Sigma)$**:** Parse the signature Σ as in (9) and C_{CS} as $(C_1, C_2, C_z, C_\sigma, C_{\mathsf{ID}})$. Then, output 1 if the the zero-knowledge proof verifies. Namely,

1. Compute the group elements R_1, R_2, $R_3 \in \mathbb{G}$ as:
$$R_1 = g^{s_\theta} \cdot C_1^{-c}, \quad R_2 = h^{s_\theta} \cdot C_2^{-c}, \quad R_3 = v^{s_{\mathsf{ID}}} \cdot X_{\mathsf{ID}}^{s_\theta} \cdot C_{\mathsf{ID}}^{-c}; \quad (10)$$
and the element $R_4 \in \mathbb{G}_T$ as
$$\big(e(X_z, \hat{g}_z) \cdot e(X_\sigma, \hat{g}_1)\big)^{s_\theta} \cdot \big(e(\tilde{\sigma}_2, \hat{g}_2) \cdot e(\tilde{\sigma}_3, \hat{g}_4)\big)^{-s_{\mathsf{ID}}} \cdot \big(e(C_z, \hat{g}_z) \cdot e(C_\sigma, \hat{g}_1) \cdot e(\tilde{\sigma}_2, \hat{g}_3) \cdot e(\tilde{\sigma}_3, \hat{g}_5)\big)^{-c}. \quad (11)$$
2. Return 1 if $c = H(M, C_{\mathsf{CS}}, \tilde{\sigma}_2, \tilde{\sigma}_3, R_1, R_2, R_3, R_4)$ and 0 otherwise.

Open$(\mathcal{Y}, \mathcal{S}_{\mathsf{OA}}, M, \Sigma)$**:** Given a pair (M, Σ) and the OA's private key $S_{\mathsf{OA}} = \big(x_z, y_z, x_\sigma, y_\sigma, x_{\mathsf{ID}}, y_{\mathsf{ID}}\big)$:

1. Decrypt $C_{\mathsf{CS}} = (C_1, C_2, C_z, C_\sigma, C_{\mathsf{ID}})$ by computing $\sigma_1 = C_\sigma \cdot C_1^{-x_\sigma} \cdot C_2^{-y_\sigma}$, $\pi = C_z \cdot C_1^{-x_z} \cdot C_2^{-y_z}$ and $V_{\mathsf{ID}} = C_{\mathsf{ID}} \cdot C_1^{-x_{\mathsf{ID}}} C_2^{-y_{\mathsf{ID}}}$.
2. Search V_{ID} in the database of joining transcripts (8) and check that it corresponds to a valid signature $\big(\tilde{\sigma}_1, \tilde{\sigma}_2, \tilde{\sigma}_3, \tilde{\pi}\big)$ for the committed value V_{ID}. If so, return the corresponding i, otherwise return $\perp$.

It is possible to spare one group element in the signature by eliminating the encryption C_{ID} of v^{ID} which is only used to open signatures in constant time. Then, the opening algorithm has to check for each transcript if $(\tilde{\sigma}_1, \tilde{\sigma}_2, \tilde{\sigma}_3, \tilde{\pi})$ corresponds to the identifier ID embedded in $(\sigma_1, \hat{G}_{2,\mathsf{ID}}, \hat{G}_{4,\mathsf{ID}})$ by testing the relation

$$1 \stackrel{?}{=} e(\tilde{\pi}, \hat{g}_z) \cdot e(\tilde{\sigma}_1, \hat{g}_1) \cdot e(\tilde{\sigma}_2, \hat{G}_{2,\mathsf{ID}} \cdot \hat{g}_3) \cdot e(\tilde{\sigma}_3, \hat{G}_{4,\mathsf{ID}} \cdot \hat{g}_5) \cdot e(\Omega, \hat{g}_6).$$

This results in a modified opening algorithm which takes $O(N)$ in the worst-case. In applications where signature openings are infrequent, this is acceptable.

5.1 Security

THEOREM 3. *If SXDH holds in $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$, the scheme is CCA-anonymous in the random oracle model.*

PROOF. We use a sequence of games where, for each i, W_i is the event that the adversary $\mathcal{A}$ wins in Game i. At the first transition, we need to rely on the security of the computational soundness of the QA-NIZK argument of Section 2.2 which relies on the SXDH assumption, since $\tilde{\sigma}_2$ and $\tilde{\sigma}_3$ appear un-encrypted in each group signature.

Game 0: This is the real CCA-anonymity game. In the challenge phase, the adversary outputs two valid membership certificates and membership secrets $(\mathsf{cert}_0^\star, \mathsf{sec}_0^\star), (\mathsf{cert}_1^\star, \mathsf{sec}_1^\star)$ and obtains a challenge signature which the challenger computes using $(\mathsf{cert}_d^\star, \mathsf{sec}_d^\star)$, where $d \stackrel{R}{\leftarrow} \{0,1\}$. We define W_0 to be the event that the adversary outputs $d' = d$.

Game 1: This game is as Game 0, except that the challenger $\mathcal{B}$ aborts in the event, which we call F_1, that $\mathcal{A}$ chooses membership certificates $\mathsf{cert}_0^\star, \mathsf{cert}_1^\star$ for which one of the underlying signatures $(\sigma_1^\star, \sigma_2^\star, \sigma_3^\star, \pi^\star)$ correctly verifies but $\log_g(\sigma_2^\star) \neq \log_h(\sigma_3^\star)$. This implies that the vector $(\sigma_1^\star, \sigma_2^{\star\mathsf{ID}}, \sigma_2^\star, \sigma_3^{\star\mathsf{ID}}, \sigma_3^\star, \Omega)$ is outside the row space of the matrix $\mathbf{M}$ (1), so that F_1 would contradict the soundness of the QA-NIZK proof of [38] (via the same arguments as in Theorem 9 of [41] since the matrix can be witness-samplable here) and thus the DDH assumption in $\hat{\mathbb{G}}$. We have $[\Pr[W_1] - P[W_0]| \leq \mathbf{Adv}_{\hat{\mathbb{G}}}^{\mathrm{DDH}}(\lambda)$.

Game 2: We change the way to generate the challenge signature $\Sigma^\star$. Instead of faithfully running the Schnorr-like protocol, we use the HVZK-simulator to produce the proofs $s_\theta, s_{\mathsf{ID}}$ without knowing the witnesses θ, ID. Namely, we pick c, s_θ, $s_{\mathsf{ID}} \stackrel{R}{\leftarrow} \mathbb{Z}_p$ at random and set $R_1 = g^{s_\theta} \cdot C_1^{-c}$, $R_2 = h^{s_\theta} \cdot C_2^{-c}$, $R_3 = v^{s_{\mathsf{ID}}} \cdot X_{\mathsf{ID}}^{s_\theta} \cdot C_{\mathsf{ID}}^{-c}$ as well as $R_4 \in \mathbb{G}_T$ as in (11). Then, we program the random oracle and assign the output c to the hash value $H(M, C_{\mathsf{CS}}, \tilde{\sigma}_2, \tilde{\sigma}_3, R_1, R_2, R_3, R_4)$. In the unlikely event that this value was previously defined (which only happens with probability at most $1/p^3$), the challenger aborts. Thus $|\Pr[W_2] - \Pr[W_1]| \leq 1/p^3$

Game 3: We modify again the generation of the challenge signature $\Sigma^\star$. Namely, the challenger computes $C_z, C_\sigma, C_{\mathsf{ID}}$ using $\mathcal{S}_{\mathsf{OA}}$ as follows

$$C_z = \tilde{\pi} \cdot C_1^{x_z} \cdot C_2^{y_z},$$
$$C_\sigma = \tilde{\sigma} \cdot C_1^{x_\sigma} \cdot C_2^{y_\sigma}, \quad C_{\mathsf{ID}} = v^{\mathsf{ID}} \cdot C_1^{x_{\mathsf{ID}}} \cdot C_2^{y_{\mathsf{ID}}}.$$

The distribution of $(C_z, C_\sigma, C_{\mathsf{ID}})$ remains the same and we have $\Pr[W_3] = \Pr[W_2]$.

Game 4: Here, we modify the distribution of the challenge signature and replace $C_2 = h^\theta$ by $C_2 = h^{\theta+\theta'}$, for a randomly chosen $\theta' \stackrel{R}{\leftarrow} \mathbb{Z}_p$. We prove in Lemma 1 that $|\Pr[W_4] - \Pr[W_3]| \leq \mathbf{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(\lambda)$.

Game 5: We introduce one more change. Instead of sampling $h \in_R \mathbb{Z}_p$, the challenger chooses a random $\alpha \stackrel{R}{\leftarrow} \mathbb{Z}_p$ at the beginning of the game, sets $h = g^\alpha$ and retains the information $\alpha = \log_g(h)$ (note that we are done with the DDH assumption and we can henceforth use $\alpha = \log_g(h)$). At each signature opening query, the challenger returns $\perp$ on any signature $\Sigma = (C_1, C_2, C_z, C_\sigma, C_{\mathsf{ID}}, \tilde{\sigma}_2, \tilde{\sigma}_3, c, s_{\mathsf{ID}}, s_\theta)$ such that $C_2 \neq C_1^\alpha$. Game 5 remains the same as Game 4. until the event E_5 that $\mathcal{A}$ queries the opening of a signature that properly verifies although $C_2 \neq C_1^\alpha$. Lemma 2 states that $\Pr[E_5] \leq q_O \cdot q_H / p$, where q_O is the number of opening queries and q_H is the number of random oracle queries.

In Game 5, $\Sigma^\star$ perfectly hides $(\tilde{\pi}, \tilde{\sigma}_1, v^{\mathsf{ID}})$. Indeed,

$$C_1 = g^\theta, \quad C_2 = h^{\theta+\theta'}, \quad C_z = (\tilde{z} \cdot h^{\theta' \cdot y_z}) \cdot X_z^\theta,$$
$$C_\sigma = (\tilde{\sigma}_1 \cdot h^{\theta' \cdot y_\sigma}) \cdot X_\sigma^\theta, \quad C_{\mathsf{ID}} = (v^{\mathsf{ID}} \cdot h^{\theta' \cdot y_{\mathsf{ID}}}) \cdot X_{\mathsf{ID}}^\theta$$

and $(y_\sigma, y_z, y_{\mathsf{ID}}) \in \mathbb{Z}_p^3$ are completely independent of $\mathcal{A}$'s view. The only way for $\mathcal{A}$ to infer information about $(y_\sigma, y_z, y_{\mathsf{ID}})$ is to make opening queries on signatures such that $C_2 \neq C_1^\alpha$. However, all such signatures are declared invalid in Game 5. It comes that $\Pr[W_5] = 1/2$.

Finally, $\mathcal{A}$'s advantage $\left|\Pr[W_0] - 1/2\right|$ is bounded by

$$\mathbf{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(\lambda) + \mathbf{Adv}_{\hat{\mathbb{G}}}^{\mathrm{DDH}}(\lambda) + \frac{q_O \cdot q_H}{p} + \frac{1}{p^3},$$

which concludes the proof. □

LEMMA 1. *In Game 4, the adversary $\mathcal{A}$ wins the anonymity game with negligibly different probabilities than in Game 3 if the DDH assumption holds in $\mathbb{G}$.*

PROOF. Let us assume that an adversary $\mathcal{A}$ wins with noticeably different probabilities in Game 4 and Game 3. We then construct a DDH distinguisher $\mathcal{B}$ from $\mathcal{A}$.

Our reduction $\mathcal{B}$ takes as input a DDH instance (g^a, g^b, η), where $\eta = g^{a(b+c)}$ and has to decide with non-negligible probability ε whether $c = 0$ or $c \in_R \mathbb{Z}_p$. To achieve this, $\mathcal{B}$ sets $h = g^a$ and computes the challenge signature as $C_1 = g^b$ and $C_2 = \eta$. The rest of the game continues like in Game 3 (which is also the same as in Game 2). If $\mathcal{A}$ wins and correctly guesses $d' = d \in \{0,1\}$, $\mathcal{B}$ outputs 1, meaning that $C_2 = h^b = g^{ab}$. Otherwise, $\mathcal{B}$ returns 0 meaning that $(g^a, g^b, \eta) \in_R \mathbb{G}^3$.
It is easy to see that $\mathcal{B}$'s advantage as a DDH distinguisher is ε if $|\Pr[W_4] - \Pr[W_3]| = \varepsilon$. □

LEMMA 2. *In Game 5, we have $\Pr[E_5] \leq q_O \cdot q_H / p$.*

PROOF. This proof uses idea similar to the security proof of the Katz-Wang [37] signature scheme. In Game 5, event E_5 happens if $\log_g(C_1) \neq \log_h(C_2)$ and the verification equations (10) and (11) holds. In particular, we

have $R_1 = g^{s_\theta} \cdot C_1^{-c}$ and $R_2 = h^{s_\theta} \cdot C_2^{-c}$, which can be interpreted as a linear system with unknowns $(c, s_\theta) \in \mathbb{Z}_p^2$

$$\begin{cases} \log_g(R_1) = s_\theta - \log_g(C_1) \cdot c \mod p, \\ \log_h(R_2) = s_\theta - \log_h(C_2) \cdot c \mod p. \end{cases} \quad (12)$$

We can assume w.l.o.g. that each opening query is preceded by the corresponding random oracle query (otherwise, the reduction can simply make the hash query for itself). The input of each hash query contains a pair (R_1, R_2) determining the non-homogeneous terms of the linear system (12). Since $\log_g(C_1) \neq \log_h(C_2)$, the system is full-rank, so that for each (R_1, R_2), there is exactly one pair $(c, s_\theta) \in \mathbb{Z}_p^2$ that satisfies (12). The probability that, in response to a random oracle query, the reduction returns the value of c which is uniquely determined by (12) is at most $1/p$. For all hash queries, the probability that one of them be answered with the uniquely determined $c \in \mathbb{Z}_q$ is *at most* q_H/p. A union bound over all opening queries implies that the probability that the event E_4 happens is smaller than $\Pr[E_4] \leq q_O \cdot q_H/p$. □

The proof of security against misidentification attacks requires the reduction to rewind a the proof of knowledge of ID at each execution of the join protocol with the adversary attempting to escape traceability. For this reason, we need to assume that users join the system sequentially, rather than concurrently. However, this problem can be solved as in [27] by having the user send an extractable commitment to ID and non-interactively prove (via the Fiat-Shamir heuristic) that he did so correctly. This allows the reduction to extract ID without rewinding the user at each execution of Join. Then, the proof of security against framing attacks must be modified by having the reduction simulate the proof of knowledge of ID (by programming a random oracle) and rely on the hiding property of the extractable commitment.

THEOREM 4. *In the ROM, the scheme is secure against mis-identification attacks under the SXDH assumption in* $(\mathbb{G}, \hat{\mathbb{G}})$.

PROOF. The proof uses the forking technique [47] which consists in implicitly rewinding the zero-knowledge proof by running the adversary twice and changing the outputs of the random oracle after the hash query that involves the forgery message. The Forking Lemma [47] – more precisely, its generalization given by Bellare and Neven [10] – ensures that, after two runs of the adversary, the reduction can extract witnesses of which knowledge is demonstrated by the signature of knowledge.

Let us assume an attacker $\mathcal{A}$ against the misidentification game that wins with non-negligible probability ε. We build an adversary $\mathcal{B}$ against the chosen-message security of the signature scheme of section 3.

Keygen. At the key generation, $\mathcal{B}$ invokes its own challenger for the chosen-message security game to obtain the public key pk_s for the signature scheme. pk_s is embedded in the group public key $\mathcal{Y}$. Except for $\mathcal{S}_{\mathsf{GM}}$, all keys are generated as in the normal Keygen algorithm.

Join. To answer joining queries without knowing sk_s, $\mathcal{B}$ uses the knowledge extractor of the proof of knowledge of $\mathsf{ID} = \log_v(V_{\mathsf{ID}})$ to extract the identity to be signed. Namely, on a Join query, the reduction $\mathcal{B}$ rewinds the adversary $\mathcal{A}$ in order to extract the witness $\mathsf{ID} = \log_v(V_{\mathsf{ID}})$ of which $\mathcal{A}$ demonstrates knowledge at step 3 of the join protocol. Having extracted $\mathsf{ID} \in \mathbb{Z}_p$, $\mathcal{B}$ invokes its own signing oracle on the message ID to obtain $(\sigma_1, \sigma_2, \sigma_3, z, r)$. Then, $\mathcal{B}$ returns $\mathsf{cert}_i = (i, V_{\mathsf{ID}}, \sigma_1, \sigma_2, \sigma_3, z, r)$ as in a normal execution of the join protocol.

At some point, the attacker $\mathcal{A}$ produces a valid forgery $(M^\star, \Sigma^\star = (C_1^\star, C_2^\star, C_z^\star, C_\sigma^\star, C_{\mathsf{ID}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, c^\star, s_{\mathsf{ID}}^\star, s_\theta^\star))$ for which the opening algorithm does not reveal a properly registered identity. With all but negligible probability, $\mathcal{A}$ must have queried the random oracle value $H(M^\star, C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, R_1^\star, R_2^\star, R_3^\star, R_4^\star)$ which would have been unpredictable otherwise.

Thus, $\mathcal{B}$ replays the adversary $\mathcal{A}$ with the *same* input and random tape as in the first run. In the second run, the random oracle is also the same until the hash query $H(M^\star, C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, R_1^\star, R_2^\star, R_3^\star, R_4^\star)$. At this point, the forking occurs and $\mathcal{B}$ outputs fresh random oracle values. By the Forking Lemma of [10], $\mathcal{B}$ obtains two suitably related forgeries with non-negligible probability $\varepsilon \cdot (\varepsilon/q_H - 1/p)$. Namely, $\mathcal{B}$ will obtain two matching transcripts $(C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, c^\star, s_{\mathsf{ID}}^\star, s_\theta^\star)$, $(C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, c^\dagger, s_{\mathsf{ID}}^\dagger, s_\theta^\dagger)$ of the Σ protocol for the commitment message $\mathsf{com} = (C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, R_1^\star, R_2^\star, R_3^\star, R_4^\star)$. From the responses $s_{\mathsf{ID}}^\star$ and $s_{\mathsf{ID}}^\dagger$ (that necessarily involve the same identifier $\mathsf{ID}^\star$ which is uniquely determined by $C_{\mathsf{CS}}^\star = (C_1^\star, C_2^\star, C_z^\star, C_\sigma^\star, C_{\mathsf{ID}}^\star)$), $\mathcal{B}$ runs the knowledge extractor of to obtain $\mathsf{ID}^\star \in \mathbb{Z}_p$. Namely, given $(c^\star, c'^\star, s_\theta^\star, s_\theta'^\star, s_{\mathsf{ID}}^\star, s_{\mathsf{ID}}'^\star) \in \mathbb{Z}_p^6$ with

$$c^\star \neq c^\dagger, \qquad s_\theta^\star \neq s_\theta^\dagger \qquad s_{\mathsf{ID}}^\star \neq s_{\mathsf{ID}}^\dagger$$

which verifies the relation (10) , (11) for the same commitment $(R_1^\star, R_2^\star, R_3^\star, R_4^\star) \in \mathbb{G}^4$, one can compute the secrets $\mathsf{ID}^\star = \frac{s_{\mathsf{ID}}^\dagger - s_{\mathsf{ID}}^\star}{c^\star - c^\dagger} \bmod p$ and $\theta^\star = \frac{s_\theta^\dagger - s_\theta^\star}{c^\star - c^\dagger} \bmod p$.

Finally $\mathcal{B}$ uses $\mathcal{S}_{\mathsf{OA}}$ to extract $\tilde{\sigma}_1^\star, \tilde{r}^\star, \tilde{z}^\star$ and outputs $\big(\mathsf{ID}^\star, \sigma^\star = (\tilde{\sigma}_1^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, \tilde{r}^\star, \tilde{z}^\star)\big)$ as a forgery for the signature scheme of Section 3. □

THEOREM 5. *In the ROM, the scheme is secure against framing attacks under the SDL assumption*

PROOF. Let us assume that a PPT adversary $\mathcal{A}$ can create, with advantage ε, a forgery $(M^\star, \sigma^\star)$ that opens to some honest user $i \in U^b$ who did not sign $M^\star$. We give a reduction $\mathcal{B}$ that uses $\mathcal{A}$ to break SDL.

Algorithm $\mathcal{B}$ takes as input an SDL instance $(g, \hat{g}, g^a, \hat{g}^a)$ and uses its interaction with the adversary $\mathcal{A}$ to compute $a \in \mathbb{Z}_p$. To generate the group public key $\mathcal{Y}$, $\mathcal{B}$ runs all the steps of the real setup algorithm Keygen except step 1. At step 1, $\mathcal{B}$ defines the generators $g, \hat{g}$ in pk_s to be those of its input and computes $h = g^{\alpha_h}$, $v = g^{\alpha_v}$, $w = g^{\alpha_w}$, $\hat{g}_z = \hat{g}^{\alpha_z}$ for randomly chosen scalars $\alpha_h, \alpha_v, \alpha_w, \alpha_z \xleftarrow{R} \mathbb{Z}_p$. In order to compute $\{z_j\}_{j=1}^3$ of crs contained in pk_s, $\mathcal{B}$ chooses $\mathsf{tk} = \{\chi_j\}_{j=1}^6$ of step 4 of the key generation algorithm of the signature scheme of Section 3 with $\ell = 1$. (Note that when $\ell = 1$, $n = 6$ and that $\{z_j\}_{j=1}^3$ are QA-NIZK argument for the vectors $(g, 1, 1, 1, 1, h)$, $(v, g, 1, h, 1, 1)$ and $(w, 1, g, 1, h, 1)$. Moreover $\{\hat{g}_i = \hat{g}_z^{\chi_i}\}_{i=1}^6$ are the verifying key.) As a result of this setup phase, $\mathcal{B}$ knows $\mathcal{S}_{\mathsf{GM}} = \mathsf{sk}_s = \omega$, $\mathcal{S}_{\mathsf{OA}} = \big(x_z, y_z, x_\sigma, y_\sigma, x_{\mathsf{ID}}, y_{\mathsf{ID}}\big)$ and even tk. The adversary $\mathcal{A}$ is run on input of the group public key $\mathcal{Y} := (\mathsf{pk}_s, (X_z, X_\sigma, X_{\mathsf{ID}}), H)$, which has the same

distribution as in the real attack game.

Should $\mathcal{A}$ decide to corrupt the group manager or the opening authority during the game, $\mathcal{B}$ is able to reveal $\mathcal{S}_{\mathsf{GM}} = \mathsf{sk}_s$ and $\mathcal{S}_{\mathsf{OA}}$ when requested. In addition, $\mathcal{B}$ must be able to answer the following queries.

- $Q_{\mathsf{b\text{-}join}}$-queries: At any time $\mathcal{A}$ can act as a corrupted group manager and introduce a new honest user i in the group by invoking the $Q_{\mathsf{b\text{-}join}}$ oracle. Then, $\mathcal{B}$ runs $\mathsf{J}_{\mathsf{user}}$ on behalf of the honest user in an execution of Join. At step 1 of Join, $\mathcal{B}$ picks a random $\delta_i \xleftarrow{R} \mathbb{Z}_p$ and uses tk to compute the tuple $(V_i, Z_i, \hat{G}_{2,i}, \hat{G}_{4,i})$, for an unknown $\mathsf{sec}_i = \mathsf{ID}_i = a \cdot \delta_i \in \mathbb{Z}_p$, that J_{GM} expects at step 1 of the join protocol. Namely, $\mathcal{B}$ computes the vector $\vec{v}_i = (V_i, G_i, 1, H_i, 1, 1) = (v, g, 1, h, 1, 1)^{\mathsf{ID}_i}$ as

$$V_i = (g^a)^{\alpha_v \cdot \delta_i}, \quad G_i = (g^a)^{\delta_i}, \quad H_i = (g^a)^{\alpha_h \cdot \delta_i},$$

and then computes Z_i as a simulated QA-NIZK proof for $\vec{v}_i \in \mathbb{G}^6$ using tk. A straightforward calculation shows that $Z_i = z_2^{\mathsf{ID}_i}$ since the QA-NIZK argument of Section 2.2 has a deterministic proving algorithm, so that $(V_i, Z_i, \hat{G}_{2,i}, \hat{G}_{4,i})$ successfully passes the test of step 2. As for the last two components, for each $j \in \{2, 4\}$, $\mathcal{B}$ computes

$$\hat{G}_{j,i} := (\hat{g}^a)^{\delta_i(\alpha_z \chi_j + \alpha_r \gamma_j)} = (\hat{g}_z^{\chi_j} \hat{g}_r^{\gamma_j})^{\mathsf{ID}_i} = \hat{g}_j^{\mathsf{ID}_i},$$

At step 3 of Join, $\mathcal{B}$ simulates the interactive proof of knowledge of $\mathsf{ID}_i = \log_v(V_i)$ using the simulator. In the rest of the protocol, $\mathcal{B}$ proceeds like the actual run and obtains $\mathsf{cert}_i = (i, V_i, \sigma_1, \sigma_2, \sigma_3, \pi)$. Finally, $\mathcal{B}$ stores $(\mathsf{cert}_i, Z_i, \delta_i, \hat{G}_{2,i}, \hat{G}_{4,i})$.

- Q_{sig}-queries: When $\mathcal{A}$ requests user $i \in U^b$ to sign a message M, $\mathcal{B}$ is able to use the membership certificate $\mathsf{cert}_i = (i, V_i, \sigma_1, \sigma_2, \sigma_3, \pi)$ to compute the ciphertext C_{CS} at steps 1-2 of the signing algorithm. While $\mathcal{B}$ does not know the witness $\mathsf{ID}_i = a \cdot \delta_i \in \mathbb{Z}_p$ to generate a proof at step 3, $\mathcal{B}$ is able to simulate the non-interactive proof $(c, s_{\mathsf{ID}}, s_\theta)$, for a randomly chosen challenge $c \xleftarrow{R} \mathbb{Z}_p$ by programming the random oracle. More precisely, $\mathcal{B}$ re-randomizes the certificate cert_i by picking $r \xleftarrow{R} \mathbb{Z}_p$ and computing

$$\tilde{\sigma}_1 = \sigma_1 \cdot (V_i \cdot w)^r \qquad \tilde{\sigma}_2 = \sigma_2 \cdot g^r,$$
$$\tilde{\pi} = \pi \cdot (Z_i \cdot z_3)^r, \qquad \tilde{\sigma}_3 = \sigma_3 \cdot h^r.$$

Then $\mathcal{B}$ encrypts $\tilde{\pi}$, $\tilde{\sigma}_1$ and V_i as in the real signing algorithm to get $C_{\mathsf{CS}} = (C_1, C_2, C_z, C_\sigma, C_{\mathsf{ID}})$. Then, $\mathcal{B}$ chooses $c, s_{\mathsf{ID}}, s_\theta \in \mathbb{Z}_p$ and computes R_1, R_2, R_3, R_4 as in (10) and (11) of Verify. Finally, $\mathcal{B}$ programs H to return c on inputs $(M, C_{\mathsf{CS}}, \tilde{\sigma}_2, \tilde{\sigma}_3, R_1, R_2, R_3, R_4)$. In the event that H is already defined at that point, $\mathcal{B}$ aborts. The probability to fail at one signing query is $\leq q_s/p^3$, where q_s is the number of signing queries.

When $\mathcal{A}$ halts, it presumably frames some honest user $i^\star \in U^b$ by outputting a signature

$$\Sigma^\star = (C_1^\star, C_2^\star, C_z^\star, C_\sigma^\star, C_{\mathsf{ID}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, c^\star, s_{\mathsf{ID}}^\star, s_\theta^\star),$$

for some message $M^\star$, that opens to $i^\star \in U^b$ although user $i^\star$ did not sign $M^\star$. With high probability, $\mathcal{A}$ must have queried the hash value $H(M^\star, C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, R_1^\star, R_2^\star, R_3^\star, R_4^\star)$, which would be unpredictable otherwise. Hence, $\mathcal{B}$ can run $\mathcal{A}$ a second time with the same input and random tape. At the moment when $\mathcal{A}$ queries $H(M^\star, C_{\mathsf{CS}}^\star, \tilde{\sigma}_2^\star, \tilde{\sigma}_3^\star, R_1^\star, R_2^\star, R_3^\star, R_4^\star)$ in the second run, $\mathcal{B}$ starts responding with different random oracle values which depart from those of the initial run. The Forking Lemma of [10] ensures that, with non-negligible probability the second run will result in a forgery $\Sigma^\dagger = (C_1^\star, C_2^\star, C_z^\star, C_\sigma^\star, C_{\mathsf{ID}}^\star, \tilde{\sigma}_2^\dagger, \tilde{\sigma}_3^\dagger, c^\dagger, s_{\mathsf{ID}}^\dagger, s_\theta^\dagger)$ on the same message $M^\star$, with distinct challenges $c^\dagger \neq c^\star$. From the two responses $(s_{\mathsf{ID}}^\star, s_\theta^\star)$, $(s_{\mathsf{ID}}^\dagger, s_\theta^\dagger)$, $\mathcal{B}$ can extract witnesses $(\theta^\star, \mathsf{ID}^\star)$ satisfying $C_{\mathsf{ID}}^\star = v^{\mathsf{ID}^\star} X_{\mathsf{ID}}^{\theta^\star}$ and which identifies $V_i^\star = v^{\mathsf{ID}^\star}$. At this stage, $\mathcal{B}$ can compute and output the sought-after SDL solution $a := \mathsf{ID}^\star/\delta_i$ in $\mathbb{Z}_p$. This observation tells us that, if $\mathcal{A}$ has advantage ε as a framing adversary making q_H random oracle queries, then $\mathcal{B}$ implies an algorithm solving the SDL problem with probability $\varepsilon(\varepsilon/q_H - 1/p)$. □

We stress that the proofs can be easily adapted to the case where the opening algorithm has linear complexity in the number of users.

5.2 Comparison with Existing Schemes

Table 1 compares our scheme with previous practical group signatures based on pairing-related assumptions. Since we focus on practical schemes, we only consider those in the random oracle model. To make the comparison possible, we use 256-bit group orders, so that elements of $\mathbb{G}$ and $\mathbb{Z}_p$ are encoded using 256 bits each.

The scheme of Boneh, Boyen and Shacham [14] is the first scheme providing short signatures: each signature is comprised of 3 group elements and 6 elements of $\mathbb{Z}_p$. However, this scheme is designed for static groups only and relies on the Strong Diffie-Hellmann assumption, which is a non-standard q-type assumption, and its anonymity is only proved in the CPA sense.

Delerablée and Pointcheval [27] presented a scheme designed for a dynamically growing group and which is also fully (i.e., CCA) anonymous. The security of their scheme is based on the eXternal Diffie-Hellman assumption (XDH), which we also use here, and the q-SDH assumption. In [27], each signature consists of 4 group elements and 5 scalars in $\mathbb{Z}_p$, which leads to the same signature size as previously. They also proposed a variant to get rid of the XDH assumption at the cost of 2 more group elements and one more scalar, but they still rely on the q-SDH assumption.

Bichsel *et al.* [12] proposed a very short group signature for dynamic groups, where each signature consists of 3 group elements and 2 elements in $\mathbb{Z}_p$. The downsides are their use the LRSW assumption [42], which is a very *ad-hoc* interactive assumption, and their security notion is not fully-anonymous, but is an hybrid security with selfless-anonymity, which is marked "CCA-" in Table 1. Another caveat is that, unlike the two previous systems, the opening complexity of their scheme is linear in the number of group members.

In 2015, Pointcheval and Sanders [46] gave another instantiation of [12] based on a variant of the LRSW assumption in the asymmetric setting (meaning using only Type III pairings), which provides even shorter signatures than [12] with the same downsides. Their scheme

Name	Signature length			Assumptions	Group Type	Anonymity
	$\mathbb{G}$	$\mathbb{Z}_p$	bits			
Ours	7	3	2560 bits	SXDH + SDL	Dynamic	CCA
Boneh-Boyen-Shacham	3	6	2304 bits	SDH + DLIN	Static	CPA
Delerablée-Pointcheval	4	5	2304 bits	SDH + XDH	Dynamic	CCA
Bichsel *et al.*	3	2	1280 bits	LRSW + SDL	Dynamic	CCA-
Pointcheval-Sanders	2	2	1024 bits	LRSW	Dynamic	CCA-

Table 1: Comparison between different group signature schemes

provides signatures composed of only 2 group elements in $\mathbb{G}$ and 2 scalars in $\mathbb{Z}_p$.

Our main contribution compared to these schemes is to provide size-comparable signatures – we recall that our scheme is composed of 7 group elements and 3 scalars in $\mathbb{Z}_p$ – while relying on standard, constant-size assumptions. Moreover, we can notice that we can save one element in $\mathbb{G}$ at the expense of a linear-time opening algorithm in the number N of group users (like [12]).

Acknowledgements

The first author was supported by the "Programme Avenir Lyon Saint-Etienne de l'Université de Lyon" in the framework of the programme "Investissements d'Avenir" (ANR-11-IDEX-0007). The third author was supported in part by the European Research Council (FP7/2007-2013 Grant Agreement no. 339563 – CryptoCloud) and by the F.R.S.-F.N.R.S. in Belgium.

6. REFERENCES

[1] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo. Structure-Preserving Signatures and Commitments to Group Elements. In *Crypto'10*, pp. 209–236, 2010.

[2] N. Akagi, Y. Manabe, T. Okamoto. An Efficient Anonymous Credential System. In *FC*'08, pp. 272–286, 2008.

[3] G. Ateniese, J. Camenisch, M. Joye, G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Crypto'00*, pp. 255–270, 2000.

[4] M.-H. Au, W. Susilo, Y. Mu. Constant-Size Dynamic k-TAA. In *SNC*'06, pp. 111–125, 2006.

[5] N. Baric, B. Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In *Eurocrypt*'97, pp. 480-494, 1997.

[6] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, H. Shacham. Randomizable Proofs and Delegatable Anonymous Credentials In *Crypto'09*, pp. 108-125, 2009.

[7] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. P-signatures and non-interactive anonymous credentials. In *TCC'08*, pp. 356-374, 2008.

[8] M. Bellare, A. Boldyreva, K. Kurosawa, J. Staddon. Multirecipient Encryption Schemes: How to Save on Bandwidth and Computation Without Sacrificing Security. *IEEE Trans. on Information Theory* 53(11), pp. 3927–3943, 2007.

[9] M. Bellare, D. Micciancio, B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *Eurocrypt'03*, pp. 614–629, 2003.

[10] M. Bellare and G. Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *CCS'06*, pp. 390-399, ACM, 2006.

[11] M. Bellare, H. Shi, C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA'05*, pp. 136–153, 2005.

[12] P. Bichsel, J. Camenisch, G. Neven, N. P. Smart, B. Warinschi. Get Shorty via Group Signatures without Encryption. In *SCN'10*, pp. 381-398, 2010.

[13] D. Boneh, X. Boyen. Short Signatures Without Random Oracles. In *Eurocrypt'04*, pp. 56–73, 2004.

[14] D. Boneh, X. Boyen, H. Shacham. Short Group Signatures. In *Crypto'04*, pp. 41-55, 2004.

[15] J. Camenisch, M. Dubovitskaya, G. Neven. Oblivious transfer with access control. In *ACM-CCS*'09, pp. 131–140, 2009.

[16] J. Camenisch, S. Hohenberger, A. Lysyanskaya. Compact E-Cash. In *Eurocrypt'05*, pp. 302-321, 2005.

[17] J. Camenisch, A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Eurocrypt'01*, pp. 93-118, 2001.

[18] J. Camenisch , A. Lysyanskaya. A Signature Scheme with Efficient Protocols. In *SCN'04*, pp. 268-289, 2002.

[19] J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Crypto'04*, pp. 56-72, 2004.

[20] J. Camenisch, G. Zaverucha. Private Intersection of Certified Sets. In *FC*'09, pp. 108-127, 2009.

[21] D. Chaum. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. Commun. ACM 28(10), pp. 1030–1044, 1985.

[22] D. Chaum, E. van Heyst. Group Signatures. In *Eurocrypt'91*, pp. 257–265, 1991.

[23] R. Cramer. Modular Design of Secure, yet Practical Cryptographic Protocols. PhD Thesis, University of Amsterdam, 1996.

[24] R. Cramer, I. Damgård, P. MacKenzie.Efficient Zero-Knowledge Proofs of Knowledge Without Intractability Assumptions. In *PKC'00*, pp. 354-372, 2000.

[25] R. Cramer, V. Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *Crypto'98*, pp. 13-25, 1994.

[26] I. Damgård. Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In *Eurocrypt'00*, pp. 418-430, 2000.

[27] C. Delerablée and D. Pointcheval. Dynamic Fully Anonymous Short Group Signatures. In *Vietcrypt'06*, pp. 193-210, 2006.

[28] A. Fiat, A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto'86*, pp. 186-194, 2003.

[29] D. Freeman. Converting Pairing-Based Cryptosystems from Composite-Order Groups to Prime-Order Groups. In *Eurocrypt*'10, pp. 44–61, 2010.

[30] J. Garay, P. MacKenzie, K. Yang. Strengthening Zero-Knowledge Protocols Using Signatures. In *Eurocrypt'03*, pp. 177-194, 2003.

[31] J. Groth. Fully anonymous group signatures without random oracles. In *Asiacrypt'07*, pp. 164–180, 2007.

[32] J. Groth, A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Eurocrypt'08*, pp. 415–432, 2008.

[33] M. Gerbush, A. Lewko, A. O'Neill, B. Waters. Dual Form Signatures: An Approach for Proving Security from Static Assumptions. In *Asiacrypt'12*, pp. 25-42, 2012.
[34] C. Jutla, A. Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In *Asiacrypt '13*, pp. 1–20, 2013.
[35] C. Jutla, A. Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In *Crypto'14*, pp. 295-312, 2014.
[36] A. Kiayias, M. Yung. Secure scalable group signature with dynamic joins and separable authorities. International Journal of Security and Networks (IJSN) Vol. 1, No. 1/2, pp. 24–45, 2006.
[37] J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions In *CCS'03*, pp. 155-164, ACM, 2003.
[38] J. Kiltz and H. Wee. Quasi-Adaptive NIZK for Linear Subspaces Revisited In *Eurocrypt'15*, pp. 101-128, 2015.
[39] B. Libert, T. Peters, M. Joye, M. Yung. Linearly Homomorphic Structure-Preserving Signatures and Their Applications. In *Crypto'13*, pp. 289-307, 2013.
[40] B. Libert, T. Peters, M. Joye, M. Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In *Eurocrypt'14*, pp. 514-532, 2014.
[41] B. Libert, T. Peters, M. Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In *Crypto'15*, pp. 296-316, 2015.
[42] A. Lysyanskaya, R. L. Rivest, A. Sahai, S. Wolf. Pseudonym Systems. In *SAC'99*, pp. 184-199, 2000.
[43] T. Nakanishi, H. Fujii, Y. Hira, N. Funabiki. Revocable Group Signature Schemes with Constant Costs for Signing and Verifying. In *PKC*'09, pp. 463-480, 2009.
[44] M. Naor, O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *FOCS '97*, pp. 458–467, IEEE Press, 1997.
[45] T. Okamoto. Efficient Blind and Partially Blind Signatures Without Random Oracles. In *TCC*'06, pp. 80-99, 2006.
[46] D. Pointcheval and O. Sanders. Short Randomizable Signatures. To appear in *CT-RSA*'16, 2016.
[47] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures In *Journal of Cryptology 2000*, Volume 13, pp. 361-396, 2000.
[48] V. Shoup and R. Gennaro. Securing Threshold Cryptosystems against Chosen Ciphertext Attack. In *Eurocrypt'98*, pp. 1-16, 1998.
[49] T.-H. Yuen, S. Chow, C. Zhang, Siu-M. Yiu. Exponent-inversion Signatures and IBE under Static Assumptions. Cryptology ePrint Archive 2014: Report 2014/311.

APPENDIX

A. PROOF OF THEOREM 1

PROOF. We will proceed as in [41] to prove that the scheme of section 3 is secure under chosen-message attacks. Namely we will consider a sequence of hybrid games involving two kinds of signatures.

Type A signatures: These are real signatures:

$$\begin{aligned} \sigma_1 &= g^{\omega} \cdot (v_1^{m_1} \cdots v_\ell^{m_\ell} \cdot w)^s, & \sigma_2 &= g^s, \\ \pi &= z_1^{\omega} \cdot (z_2^{m_1} \cdots z_{\ell+1}^{m_\ell} \cdot z_{\ell+2})^s, & \sigma_3 &= h^s. \end{aligned} \tag{13}$$

Since $(\sigma_1, \sigma_2^{m_1}, \ldots, \sigma_2^{m_\ell}, \sigma_2, \sigma_3^{m_1}, \ldots, \sigma_3^{m_\ell}, \sigma_3, \Omega)$ is in the row space of $\mathbf{M}$, the QA-NIZK proof π has the same distribution as if it were computed as

$$\begin{aligned} \pi = \sigma_1^{-\chi_1} \cdot \left(\prod_{i=2}^{\ell+1} \sigma_2^{-\chi_i m_{i-1}} \right) \cdot \sigma_2^{-\chi_{\ell+2}} \cdot \\ \left(\prod_{i=\ell+3}^{2\ell+2} \sigma_3^{-\chi_i m_{i-\ell-2}} \right) \cdot \sigma_3^{-\chi_{2\ell+3}} \cdot \Omega^{-\chi_{2\ell+4}}. \end{aligned} \tag{14}$$

We also define **Type A′** signatures as a generalization of Type A signatures where only condition (13) are imposed and no restriction is given on π beyond the fact that it should be a valid homomorphic signature on vector (2).

Type B signatures: These use a random value $\omega' \in_R \mathbb{Z}_p$ instead of the secret key ω. We pick random $\omega', s, s_1 \xleftarrow{R} \mathbb{Z}_p$ and compute:

$$(\sigma_1, \sigma_2, \sigma_3) = (g^{\omega'} \cdot (v_1^{m_1} \cdots v_\ell^{m_\ell} \cdot w)^s,\ g^s,\ h^{s+s_1}),$$

The QA-NIZK proof π is computed as in (14) by using $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$. Note that Type B signatures can be generated without using $\omega \in \mathbb{Z}_p$.

We consider a sequence of games. In Game i, S_i denotes the event that $\mathcal{A}$ produces a valid signature $\sigma^\star$ on $M^\star$ such that $(M^\star, \sigma^\star)$ was not queried before, and by E_i the event that $\mathcal{A}$ produces a Type A′ signature.

Game 0: This is the real game. The challenger $\mathcal{B}$ produces a key pair $(\mathsf{sk}, \mathsf{pk})$ and sends pk to $\mathcal{A}$. Then $\mathcal{A}$ makes Q signature queries: $\mathcal{A}$ sends messages M_i to $\mathcal{B}$, and $\mathcal{B}$ answers by sending $\sigma_i = \mathsf{Sign}(\mathsf{sk}, M_i)$ to $\mathcal{A}$. Finally $\mathcal{A}$ sends a pair $(M^\star, \sigma^\star) \notin \{(M_i, \sigma_i)\}_{i=1}^{Q}$ and wins if $\mathsf{Verify}(\mathsf{pk}, \sigma^\star, M^\star) = 1$.

Game 1: We change the way $\mathcal{B}$ answers signing queries. The QA-NIZK proofs π are then computed as simulated QA-NIZK proofs using tk as in (14). These QA-NIZK proofs are thus simulated proofs for true statements, and then their distribution remains unchanged. We have $\Pr[S_1] = \Pr[S_1 \wedge E_1] + \Pr[S_1 \wedge \neg E_1]$. Lemma 3 states that the event $S_1 \wedge \neg E_1$ happens with all but negligible probability: $\Pr[S_1 \wedge \neg E_1] \leq \mathbf{Adv}^{\mathrm{DDH}}_{\hat{\mathbb{G}}}(\lambda) - 1/p$. Thus our task is now to upper-bound the probability $\Pr[S_1 \wedge E_1]$.

Game 2.k ($0 \leq k \leq Q$): In Game 2.k, the challenger returns a Type B signature for the first k queries. At the last $Q - k$ signature queries, the challenger answers a type A signature. Lemma 4 ensures that $\left|\Pr[S_{2.k} \wedge E_{2.k}] - \Pr[S_{2.(k-1)} \wedge E_{2.(k-1)}]\right|$ is smaller than $\mathbf{Adv}^{\mathrm{DDH}}_{\mathbb{G}}(\lambda) + 1/p$.

In Game 2.Q, we know that if SXDH holds, $\mathcal{A}$ can only output a type A′ forgery even if it only obtains type B signatures during the game. Nevertheless, lemma 5 shows that a type A′ forgery in Game 2.Q contradicts the DDH assumptions in $\mathbb{G}$. Therefore we have $\Pr[S_{2.Q} \wedge E_{2.Q}] \leq \mathbf{Adv}^{\mathrm{DDH}}_{\mathbb{G}}(\lambda)$. Putting the above altogether, the probability $\Pr[S_0]$ is upper-bounded by

$$\begin{aligned} \mathbf{Adv}^{\mathrm{DDH}}_{\hat{\mathbb{G}}}(\lambda) + \frac{1}{p} + Q\left(\mathbf{Adv}^{\mathrm{DDH}}_{\mathbb{G}}(\lambda) + \frac{1}{p}\right) + \mathbf{Adv}^{\mathrm{DDH}}_{\mathbb{G}}(\lambda) \\ < (Q+2) \cdot \left(\mathbf{Adv}^{\mathrm{SXDH}}_{\mathbb{G},\hat{\mathbb{G}}}(\lambda) + \frac{1}{p}\right). \quad \square \end{aligned}$$

LEMMA 3. *In **Game 1**, if the DDH assumption holds in $\hat{\mathbb{G}}$, $\mathcal{A}$ can only output a type A′ forgery.*

PROOF. Let $\mathcal{A}$ be an attacker that does not output a type A′ forgery. We will build an attacker $\mathcal{B}$ against the soundness of the Quasi-Adaptive NIZK (QA-NIZK) scheme, which security is implied from the double-pairing problem that reduces from DDH as explained in [39]. Let us define the vector $\boldsymbol{\sigma} \in \mathbb{G}^{2\ell+4}$ as

$$(\sigma_1^\star, \sigma_2^{\star m_1}, \ldots, \sigma_2^{\star m_\ell}, \sigma_2^\star, \sigma_3^{\star m_1}, \ldots, \sigma_3^{\star m_\ell}, \sigma_3^\star, \Omega) \in \mathbb{G}^{2\ell+4}.$$

If $(M^\star, \sigma^\star)$ is not a type A′ forgery, $\boldsymbol{\sigma}$ is then not in the row space of $\mathbf{M}$.

Our reduction $\mathcal{B}$ receives as input $\mathsf{cp} = (\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, p)$, a matrix $\mathbf{M}$ as in (1) and a common reference string crs (depending on the matrix) for an instance of the QA-NIZK scheme allowing to prove that vectors of dimension $2\ell+4$ are in the row space of $\mathbf{M}$. The generation of the matrix $\mathbf{M}$ fixes g, h and $\vec{v} = (v_1, \ldots, v_\ell, w) \in \mathbb{G}^{\ell+1}$. After that, $\mathcal{B}$ picks $\omega \xleftarrow{R} Z_p$ and $\hat{g} \xleftarrow{R} \hat{\mathbb{G}}$, and set $\Omega = h^\omega$. Then, the reduction $\mathcal{B}$ sends to $\mathcal{A}$ cp and the verification key:

$$\mathsf{pk} = \big(g, h, \hat{g}, \vec{v}, \omega, \mathsf{crs}\big).$$

Since $\mathcal{B}$ knows the secret key $\omega \in \mathbb{Z}_p$, it can answer all signing queries by honestly running the Sign algorithm, in particular, it does not need to know tk to do this.

When $\mathcal{A}$ halts, it outputs $(M^\star, \sigma^\star)$ where $\sigma^\star$ is not a Type A′ forgery, so that $\boldsymbol{\sigma}$ is not in the row space of $\mathbf{M}$. Therefore, outputting $\pi^\star$ constitutes a valid proof against the soundness property of the scheme, and thus implies an algorithm against DDH as in [38] since the matrix can be witness-samplable. □

LEMMA 4. *If DDH holds in $\mathbb{G}$, for each $k \in \{1, \ldots, Q\}$, $\mathcal{A}$ produces a type A′ forgery with negligibly different probabilities in* ***Game 2.k*** *and* ***Game 2.(k − 1)***.

PROOF. Let us assume there exists an index $k \in \{1, \ldots, Q\}$ and an adversary $\mathcal{A}$ that outputs a Type A′ forgery with smaller probability in Game 2.k than in Game 2.$(k-1)$. We build a DDH distinguisher $\mathcal{B}$.

Algorithm $\mathcal{B}$ takes in $(g^a, g^b, \eta) \in \mathbb{G}^3$, where $\eta = g^{a(b+c)}$, and decides if $c = 0$ or $c \in_R \mathbb{Z}_p$. To do this, $\mathcal{B}$ sets $h = g^a$. It picks $\omega, a_{v_1}, b_{v_1}, \ldots, a_{v_\ell}, b_{v_\ell}, a_w, b_w \xleftarrow{R} \mathbb{Z}_p$ and sets $\Omega = h^\omega$ as well as:

$$\forall i \in \{1, \ldots, \ell\} : \quad v_i = g^{a_{v_i}} \cdot h^{b_{v_i}}, \quad w = g^{a_w} \cdot h^{b_w}.$$

The reduction $\mathcal{B}$ also chooses $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$ and computes $\mathsf{crs} = (\{z_j\}_{j=1}^{2\ell+4}, \hat{g}_z, \{\hat{g}_i\}_{i=1}^{2\ell+4})$ as in steps 3-4 of Keygen. It then outputs $\mathsf{pk} = (g, h, \hat{g}, \vec{v}, \omega, \mathsf{crs})$.

Then, queries are answered depending on their index j:
Case $j < k$: $\mathcal{B}$ computes a Type B signature, $\sigma = (\sigma_1, \sigma_2, \sigma_3, \pi)$, using $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$ with the QA-NIZK simulator to computes π.
Case $j > k$: The last $Q - k - 1$ signing queries are computed as Type A signatures, which $\mathcal{B}$ is able to generate using the secret key $\omega \in \mathbb{Z}_p$ he knows and crs or $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$ to produces valid proofs.
Case $j = k$: In the k-th signing query $(m_1, \ldots, m_\ell)$, $\mathcal{B}$ embeds the DDH instance in the signature and simulates either Game 2.k or Game 2.$(k-1)$ depending on whether $\eta = g^{ab}$ or $\eta = g^{a(b+c)}$ for some $c \in_R \mathbb{Z}_p$. Namely, $\mathcal{B}$ computes $\sigma_2 = g^b$, $\sigma_3 = \eta$, and $\sigma_1 = g^\omega \sigma_2^{a_w + \sum_{i=1}^\ell a_{v_i} m_i} \sigma_3^{b_w + \sum_{i=1}^\ell b_{v_i} m_i}$. Then $\mathcal{B}$ simulates QA-NIZK proofs π as recalled in (14), and sends $\sigma = (\sigma_1, \sigma_2, \sigma_3, \pi)$ to $\mathcal{A}$.

If $\eta = g^{ab}$, the k-th signature σ is a Type A signature with $s = b$. If $\eta = g^{a(b+c)}$ for some $c \in_R \mathbb{Z}_p$, we have:

$$\begin{aligned}\sigma_1 &= g^\omega g^{ac \cdot (b_w + \sum_{i=1}^\ell b_{v_i} m_i)} (v_1^{m_1} \cdots v_\ell^{m_\ell} w)^b \\ &= g^{\omega'} (v_1^{m_1} \cdots v_\ell^{m_\ell} w)^b \\ \sigma_2 &= g^b, \qquad\qquad \sigma_3 = h^{b+c}\end{aligned}$$

Where $\omega' = \omega + ac \cdot (b_w + \sum_{i=1}^\ell b_{v_i} m_i)$. Since the term $b_w + \sum_{i=1}^\ell b_{v_i} m_i$ is uniform and independent of $\mathcal{A}$'s view, σ is distributed as a Type B signature if $\eta = g^{a(b+c)}$.

When $\mathcal{A}$ terminates, it outputs a couple $(m_1^\star \cdots m_\ell^\star, \sigma^\star)$ that has not been queried during the signing queries. Now the reduction $\mathcal{B}$ has to determine whether $\sigma^\star$ is a Type A′ forgery or not. To this end, it tests if the equality:

$$\sigma_1^\star = g^\omega \sigma_2^{\star a_w + \sum_{i=1}^\ell a_{v_i} m_i^\star} \sigma_3^{\star b_w + \sum_{i=1}^\ell b_{v_i} m_i^\star} \tag{15}$$

is satisfied. If it is, $\mathcal{B}$ outputs 1 to indicate that $\eta = g^{ab}$. Otherwise it outputs 0 and rather bets that $\eta \in_R \mathbb{G}$.

To see why this test allows recognizing Type A′ forgeries, we remark that $\sigma^\star$ is of the form:

$$\sigma_2^\star = g^s, \quad \sigma_3^\star = h^{s+s_1}, \quad \sigma_1^\star = g^{\omega + s_0} (v_1^{m_1^\star} \cdots v_\ell^{m_\ell^\star} w)^s,$$

and the goal of $\mathcal{B}$ is to decide whether $(s_0, s_1) = (0, 0)$ or not. We notice that $s_0 = a \cdot s_1 \cdot (b_w + \sum_{i=1}^\ell b_{v_i} \cdot m_i^\star)$ if the forgery fulfills relation (15) and we show this to only happen with probability $1/p$ for any $s_1 \neq 0$ meaning that Type B forgery passes the test with the same probability.

From the entire game and assuming a forgery which passes the test we have the following linear system:

$$\left(\begin{array}{c|c} \mathbf{I}_{\ell+1} & a \cdot \mathbf{I}_{\ell+1} \\ \hline \mathbf{0}_{\ell+1}^\top & ac \cdot (m_1 | \cdots | m_\ell | 1) \\ \hline \mathbf{0}_{\ell+1}^\top & as_1 \cdot (m_1^\star | \cdots | m_\ell^\star | 1) \end{array}\right) \cdot \begin{pmatrix} a_{v_1} \\ \vdots \\ a_{v_\ell} \\ a_w \\ b_{v_1} \\ \vdots \\ b_{v_\ell} \\ b_w \end{pmatrix} = \begin{pmatrix} \log_g(v_1) \\ \vdots \\ \log_g(v_\ell) \\ \log_g(w) \\ \omega' - \omega \\ s_0 \end{pmatrix}$$

where, $\mathbf{0}_{\ell+1}$ denotes the zero vector of length $\ell+1$ and $m_1, \ldots, m_\ell$ is the message involved in the k-th signing query. Note that the $(l+2)$-th equation is meaningless when $c = 0$ since then $\omega' = \omega$. However, even if $c \neq 0$ the information that $\mathcal{A}$ can infer about $(a_{v_1}, \ldots, a_{v_\ell}, a_w, b_{v_1}, \ldots, b_{v_\ell}, b_w) \in \mathbb{Z}_p^{2\ell+2}$ during the game amounts to the first $\ell+2$ equations of the system which is of full rank. It means that this vector is unpredictable since all the solutions of this linear system live in a sub-space of dimension at least one (actually $\ell = (2\ell+2) - (\ell+2)$). Finally, as long as $s_1 \neq 0$, the right value s_0 can only be guessed with probability $1/p$ since the last row of the matrix is independent of the others as soon as $(m_1, \ldots, m_\ell) \neq (m_1^\star, \ldots, m_\ell^\star) \neq 0$.

To conclude the proof, since $\mathcal{B}$ is able the tell apart the type of the forgery, if $\mathcal{A}$'s probability to output a forgery of some Type in Game $k-1$ (*i.e.* $c = 0$) was significantly different than in Game k (*i.e.* $c \neq 0$) then B would be able to solve the DDH problem with non-negligible. □

LEMMA 5. *In* ***Game 2.Q****, a PPT adversary outputting a type A′ forgery would contradict the DDH assumption in $\mathbb{G}$:* $\Pr[S_{2.Q} \wedge E_{2.Q}] \leq \mathbf{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(\lambda)$.

PROOF. We will build an algorithm $\mathcal{B}$ for solving the Computational Diffie Hellman problem (CDH) which is at least as hard as the DDH problem. The reduction $\mathcal{B}$ takes as input a tuple $(g, h, \Omega = h^{\omega})$ and computes g^{ω}. To generate pk, $\mathcal{B}$ picks $\hat{g} \xleftarrow{R} \hat{\mathbb{G}}$, $a_{v_1}, \ldots, a_{v_\ell}, a_w \xleftarrow{R} \mathbb{Z}_p$ and computes $v_1 = g^{a_{v_1}}$, $\ldots$, $v_\ell = g^{a_{v_\ell}}$, and $w = g^{a_w}$. Then $\mathcal{B}$ generates $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$, $\mathsf{crs} = (\{z_j\}_{j=1}^{\ell+2}, \hat{g}_z, \{\hat{g}_i\}_{i=1}^{2\ell+4})$ as in step 3-4 of the key generation algorithm, then sends the public key $pk = \big(g, h, \hat{g}, \boldsymbol{v}, \Omega = h^{\omega}, \mathsf{crs}\big)$ to $\mathcal{A}$.

$\mathcal{B}$ also retains $\mathsf{tk} = \{\chi_i\}_{i=1}^{2\ell+4}$ to handle signing queries. We recall that during the game, signing queries are answered by returning a Type B signature so that, using tk, $\mathcal{B}$ can answer all queries without knowing the $\omega = \log_h(\Omega)$ which is part of the CDH challenge.

The results of Lemma 4 implies that even if $\mathcal{A}$ only obtains Type B signatures, it will necessarily output a Type A′ forgery $\sigma^{\star} = (\sigma_1^{\star}, \sigma_2^{\star}, \sigma_3^{\star}, \pi^{\star})$ unless the DDH assumption does not hold in $\mathbb{G}$. This event thus allows $\mathcal{B}$ to compute $g^{\omega} = \sigma_1^{\star} \cdot \sigma_2^{\star - a_w - \sum_{i=1}^{\ell} a_{v_i} m_i^{\star}}$, which contradicts the DDH assumption in $\mathbb{G}$. □

B. DEFINITIONS FOR DYNAMIC GROUP SIGNATURES

In the setting of dynamic groups, the syntax of group signatures includes an interactive protocol which allows users to register as new members of the group at any time. The syntax and the security model are those defined by Kiayias and Yung (KY) [36]. Like the very similar Bellare-Shi-Zhang model [11], the KY model assumes an interactive *join* protocol whereby a prospective user becomes a group member by interacting with the group manager. This protocol provides the user with a membership certificate and a membership secret.

We denote by $N \in \mathsf{poly}(\lambda)$ the maximal number of group members. A dynamic group signature scheme consists of the following algorithms or protocols.

Setup: given a security parameter λ and a maximal number of group members $N \in \mathbb{N}$, this algorithm is run by a trusted party to generate a group public key $\mathcal{Y}$, the group manager's private key $\mathcal{S}_{\mathsf{GM}}$ and the opening authority's private key $\mathcal{S}_{\mathsf{OA}}$. Each key is given to the appropriate authority while $\mathcal{Y}$ is made public. The algorithm also initializes a public state St comprising a set data structure $St_{\mathsf{users}} = \emptyset$ and a string data structure $St_{\mathsf{trans}} = \epsilon$.

Join: is an interactive protocol between the group manager GM and a user $\mathcal{U}_i$ where the latter becomes a group member. The protocol involves two interactive Turing machines $\mathsf{J}_{\mathsf{user}}$ and J_{GM} that both take $\mathcal{Y}$ as input. The execution, denoted as $[\mathsf{J}_{\mathsf{user}}(\lambda, \mathcal{Y}), \mathsf{J}_{\mathsf{GM}}(\lambda, St, \mathcal{Y}, \mathcal{S}_{\mathsf{GM}})]$, ends with user $\mathcal{U}_i$ obtaining a membership secret sec_i, that no one else knows, and a membership certificate cert_i. If the protocol is successful, the group manager updates the public state St by setting $St_{\mathsf{users}} := St_{\mathsf{users}} \cup \{i\}$ as well as $St_{\mathsf{trans}} := St_{\mathsf{trans}} || \langle i, \mathsf{transcript}_i \rangle$.

Sign: given a membership certificate cert_i, a membership secret sec_i, a message M, it outputs a signature σ.

Verify: given a signature σ, a message M and a group public key $\mathcal{Y}$, this algorithm returns either 0 or 1.

Open: takes as input a message M, a valid signature σ w.r.t. $\mathcal{Y}$, the opening authority's private key $\mathcal{S}_{\mathsf{OA}}$ and the public state St. It outputs $i \in St_{\mathsf{users}} \cup \{\perp\}$, which is the identity of a group member or a symbol indicating an opening failure.

Each membership certificate contains a unique tag that identifies the user.

CORRECTNESS FOR DYNAMIC GROUP SIGNATURES. Following the terminology of [36], a public state St is *valid* if it can be reached from $St = (\emptyset, \epsilon)$ by a Turing machine having oracle access to J_{GM}. Also, a state St' is said to *extend* another state St if it is within reach from St.

As in [36], $\mathsf{cert}_i \leftrightharpoons_{\mathcal{Y}} \mathsf{sec}_i$, means that there exists coin tosses ϖ for J_{GM} and $\mathsf{J}_{\mathsf{user}}$ such that, for some valid public state St', the execution of $[\mathsf{J}_{\mathsf{user}}(\lambda, \mathcal{Y}), \mathsf{J}_{\mathsf{GM}}(\lambda, St', \mathcal{Y}, \mathcal{S}_{\mathsf{GM}})](\varpi)$ provides $\mathsf{J}_{\mathsf{user}}$ with $\langle i, \mathsf{sec}_i, \mathsf{cert}_i \rangle$.

Definition 4. A dynamic group signature scheme is correct if the following conditions are all satisfied:

1. In a valid state St, $|St_{\mathsf{users}}| = |St_{\mathsf{trans}}|$ always holds and two distinct entries of St_{trans} always contain certificates with distinct tag.
2. If $[\mathsf{J}_{\mathsf{user}}(\lambda, \mathcal{Y}), \mathsf{J}_{\mathsf{GM}}(\lambda, St, \mathcal{Y}, \mathcal{S}_{\mathsf{GM}})]$ is run by two honest parties following the protocol and $\langle i, \mathsf{cert}_i, \mathsf{sec}_i \rangle$ is obtained by $\mathsf{J}_{\mathsf{user}}$, then it holds that $\mathsf{cert}_i \leftrightharpoons_{\mathcal{Y}} \mathsf{sec}_i$.
3. For each $\langle i, \mathsf{cert}_i, \mathsf{sec}_i \rangle$ such that $\mathsf{cert}_i \leftrightharpoons_{\mathcal{Y}} \mathsf{sec}_i$, satisfying condition 2, it always holds that $\mathsf{Verify}\big(\mathsf{Sign}(\mathcal{Y}, \mathsf{cert}_i, \mathsf{sec}_i, M), M, \mathcal{Y}\big) = 1$.
4. For any outcome $\langle i, \mathsf{cert}_i, \mathsf{sec}_i \rangle$ of the interaction $[\mathsf{J}_{\mathsf{user}}(.,.), \mathsf{J}_{\mathsf{GM}}(., St, ., .)]$ for some valid St, if $\sigma = \mathsf{Sign}(\mathcal{Y}, \mathsf{cert}_i, \mathsf{sec}_i, M)$, then $\mathsf{Open}(M, \sigma, \mathcal{S}_{\mathsf{OA}}, \mathcal{Y}, St') = i$.

The Kiayias-Yung model [36] considers three security notions: security against mis-identification attacks, non frameability and (full) anonymity. These notions are formalized through experiments where the adversary interacts with a stateful interface $\mathcal{I}$ that maintains the following variables:

- $\mathsf{state}_{\mathcal{I}}$: is a data structure representing the state of the interface as the adversary invokes the oracles available in the attack games. It is initialized as $\mathsf{state}_{\mathcal{I}} = (St, \mathcal{Y}, \mathcal{S}_{\mathsf{GM}}, \mathcal{S}_{\mathsf{OA}}) \leftarrow \mathsf{Setup}(\lambda, N)$. It includes the (initially empty) set St_{users} of group members and a database St_{trans} storing the transcripts of previously executed join protocols.
- $n = |St_{\mathsf{users}}| < N$ is the current group cardinality.
- Sigs: is a database of signatures created by the signing oracle. Each entry consists of a triple (i, M, σ) indicating that message M was signed by user i.
- U^a: is the set of users that were introduced by the adversary in the system in an execution of Join.
- U^b: is the set of honest users that the adversary introduces, acting as a dishonest group manager.

When mounting attacks, adversaries will be granted access to the following oracles.

- Q_{pub}, Q_{keyGM} and Q_{keyOA}: when these oracles are invoked, the interface looks up $\mathsf{state}_{\mathcal{I}}$ and returns the group public key $\mathcal{Y}$, the GM's private key $\mathcal{S}_{\mathsf{GM}}$ and the opening authority's private key $\mathcal{S}_{\mathsf{OA}}$ respectively.

- $Q_{\mathsf{a\text{-}join}}$: allows the adversary $\mathcal{A}$ to introduce users *under his control* in the group. On behalf of the GM, the interface runs J_{GM} in interaction with the $\mathsf{J}_{\mathsf{user}}$-executing adversary who plays the role of the prospective user in the join protocol. If this protocol successfully ends, the interface increments n, updates St by inserting the new user n in both sets St_{users} and U^a. It also sets $St_{\mathsf{trans}} := St_{\mathsf{trans}}||\langle n, \mathsf{transcript}_n\rangle$.
- $Q_{\mathsf{b\text{-}join}}$: allows $\mathcal{A}$, acting as a corrupted group manager, to introduce new *honest* group members. The interface triggers an execution of $[\mathsf{J}_{\mathsf{user}}, \mathsf{J}_{\mathsf{GM}}]$ and runs $\mathsf{J}_{\mathsf{user}}$ in interaction with $\mathcal{A}$ who runs J_{GM}. If the protocol successfully completes, the interface increments n, adds user n to St_{users} and U^b and sets $St_{\mathsf{trans}} := St_{\mathsf{trans}}||\langle n, \mathsf{transcript}_n\rangle$. It stores the membership certificate cert_n and the membership secret sec_n in a *private* part of $\mathsf{state}_{\mathcal{I}}$.
- Q_{sig}: given M, an index i, the interface checks if the private area of $\mathsf{state}_{\mathcal{I}}$ contains a certificate cert_i and a membership secret sec_i. If no such $(\mathsf{cert}_i, \mathsf{sec}_i)$ exist or if $i \notin U^b$, the interface returns $\perp$. Otherwise, it outputs a signature σ on behalf of user i and also updates $\mathsf{Sigs} := \mathsf{Sigs}||(i, M, \sigma)$.
- Q_{open}: when this oracle is invoked on input of a valid pair (M, σ), the interface runs algorithm Open using the current state St. When S is a set of pairs of the form (M, σ), $Q_{\mathsf{open}}^{\neg S}$ denotes a restricted oracle that only applies the opening algorithm to pairs (M, σ) which are not in the set S.
- Q_{read} and Q_{write}: are used by $\mathcal{A}$ to read/write the content of $\mathsf{state}_{\mathcal{I}}$. At each invocation, Q_{read} outputs the whole $\mathsf{state}_{\mathcal{I}}$ but the public/private keys and the private part of $\mathsf{state}_{\mathcal{I}}$ where membership secrets are stored after $Q_{\mathsf{b\text{-}join}}$-queries. By using Q_{write}, $\mathcal{A}$ can modify $\mathsf{state}_{\mathcal{I}}$ at will as long as it does not remove or alter elements of St_{users}, St_{trans} or invalidate the public state St: for example, $\mathcal{A}$ is allowed to create dummy users as long as he/she does not re-use already existing certificate tags.

Security Against Mis-identification Attacks. In a mis-identification attack, the adversary is able to corrupt the opening authority using the Q_{keyOA} oracle. Moreover, he can also introduce malicious users in the group via $Q_{\mathsf{a\text{-}join}}$-queries. His purpose is to come up with a valid signature $\sigma^\star$ that does not open to any adversarially-controlled user.

Definition 5. A dynamic group signature scheme is secure against mis-identification attacks if, for any PPT adversary $\mathcal{A}$ involved in the experiment hereunder, we have $\mathbf{Adv}_{\mathcal{A}}^{\text{mis-id}}(\lambda) = \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\text{mis-id}}(\lambda) = 1\right] \in \mathsf{negl}(\lambda)$.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{mis\text{-}id}(\lambda)$
1. $\mathsf{state}_{\mathcal{I}} = (St, \mathcal{Y}, \mathcal{S}_{\mathsf{GM}}, \mathcal{S}_{\mathsf{OA}}) \leftarrow \mathsf{Setup}(\lambda, N)$;
2. $(M^\star, \sigma^\star) \leftarrow \mathcal{A}(Q_{\mathsf{pub}}, Q_{\mathsf{a\text{-}join}}, Q_{\mathsf{revoke}}, Q_{\mathsf{read}}, Q_{\mathsf{keyOA}})$;
3. If $\mathsf{Verify}(\sigma^\star, M^\star, \mathcal{Y}) = 0$, return 0;
4. $i = \mathsf{Open}(M^\star, \sigma^\star, \mathcal{S}_{\mathsf{OA}}, \mathcal{Y}, St')$;
5. If $i \notin U^a$ return 1; Otherwise return 0;

Non-Frameability. In framing attacks, the entire system is colluding against some honest user. The adversary can corrupt the group manager as well as the opening authority (via oracles Q_{keyGM} and Q_{keyOA}, respectively). He can also introduce honest group members (via $Q_{\mathsf{b\text{-}join}}$-queries), observe the system while these users sign messages and create dummy users using Q_{write}. The adversary eventually aims at framing an honest group member.

Definition 6. A dynamic group signature scheme is secure against framing attacks if, for any PPT adversary $\mathcal{A}$ involved in the experiment below, it holds that $\mathbf{Adv}_{\mathcal{A}}^{\text{fra}}(\lambda) = \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\text{fra}}(\lambda) = 1\right] \in \mathsf{negl}(\lambda)$.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\text{fra}}(\lambda)$
1. $\mathsf{state}_{\mathcal{I}} = (St, \mathcal{Y}, \mathcal{S}_{\mathsf{GM}}, \mathcal{S}_{\mathsf{OA}}) \leftarrow \mathsf{Setup}(\lambda, N)$;
2. $(M^\star, \sigma^\star) \leftarrow$ $\mathcal{A}(Q_{\mathsf{pub}}, Q_{\mathsf{keyGM}}, Q_{\mathsf{keyOA}}, Q_{\mathsf{b\text{-}join}}, Q_{\mathsf{sig}}, Q_{\mathsf{read}}, Q_{\mathsf{write}})$;
3. If $\mathsf{Verify}(\sigma^\star, M^\star, \mathcal{Y}) = 0$, return 0;
4. If $i = \mathsf{Open}(M^\star, \sigma^\star, \mathcal{S}_{\mathsf{OA}}, \mathcal{Y}, St') \notin U^b$, return 0;
5. If $\left(\bigwedge_{j \in U^b \text{ s.t. } j=i} (j, M^\star, *) \notin \mathsf{Sigs}\right)$ return 1;
6. Return 0;

Full Anonymity. Anonymity is formalized via a game involving a two-stage adversary. The first stage allows the adversary $\mathcal{A}$ to modify $\mathsf{state}_{\mathcal{I}}$ via Q_{write}-queries and open arbitrary signatures by probing Q_{open}. Then, $\mathcal{A}$ chooses a message $M^\star$ as well as two pairs $(\mathsf{sec}_0^\star, \mathsf{cert}_0^\star)$ and $(\mathsf{sec}_1^\star, \mathsf{cert}_1^\star)$, consisting of a valid membership certificate and a corresponding membership secret. The challenger flips a coin $d \leftarrow \{0,1\}$ and computes a challenge signature $\sigma^\star$ using $(\mathsf{sec}_d^\star, \mathsf{cert}_d^\star)$, which is given $\sigma^\star$ to $\mathcal{A}$ who is allowed further oracle queries throughout the second stage, but is restricted not to query Q_{open} for the challenge message-signature pair $(M^\star, \sigma^\star)$.

Definition 7. A dynamic group signature scheme is fully anonymous if, for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{\text{anon}}(\lambda) := |\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\text{anon}}(\lambda) = 1\right] - 1/2| \in \mathsf{negl}(\lambda)$.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\text{anon}}(\lambda)$
1. $\mathsf{state}_{\mathcal{I}} = (St, \mathcal{Y}, \mathcal{S}_{\mathsf{GM}}, \mathcal{S}_{\mathsf{OA}}) \leftarrow \mathsf{Setup}(\lambda)$;
2. $\left(aux, M^\star, (\mathsf{sec}_0^\star, \mathsf{cert}_0^\star), (\mathsf{sec}_1^\star, \mathsf{cert}_1^\star)\right)$ $\leftarrow \mathcal{A}(\mathsf{play}; Q_{\mathsf{pub}}, Q_{\mathsf{keyGM}}, Q_{\mathsf{open}}, Q_{\mathsf{read}}, Q_{\mathsf{write}})$;
3. If $\neg(\mathsf{cert}_b^\star \leftrightharpoons_{\mathcal{Y}} \mathsf{sec}_b^\star)$ for $b \in \{0,1\}$, return 0;
4. If $\mathsf{cert}_0^\star = \mathsf{cert}_1^\star$, return 0;
5. Picks random $d \leftarrow \{0,1\}$; $\sigma^\star \leftarrow \mathsf{Sign}(\mathcal{Y}, \mathsf{cert}_d^\star, \mathsf{sec}_d^\star, M^\star)$;
6. $d' \leftarrow \mathcal{A}(\mathsf{guess}; \sigma^\star, aux, Q_{\mathsf{pub}}, Q_{\mathsf{keyGM}}, Q_{\mathsf{open}}^{\neg\{(M^\star, \sigma^\star)\}}, Q_{\mathsf{read}}, Q_{\mathsf{write}})$;
7. If $d' = d$ then return 1;
8. Return 0;